NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

THE SPECTRA OF DES S-BOXES

by

Mathew B. Fukuzawa

June 2014

Thesis Advisor: Pantelimon Stănică
Second Reader: Craig Rasmussen

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June-20-2014	Master's Thesis July-02-2012 to June-20-2014

4. TITLE AND SUBTITLE

THE SPECTRA OF DES S-BOXES

5. FUNDING NUMBERS

6. AUTHOR(S)

Mathew B. Fukuzawa

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of the Navy

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

We typically do not associate the field of graph theory with the field of cryptography. In graph theory, the aim is to model relationships with a graph and examine properties of that graph. The goal of cryptography is to design a communication system over a nonsecure channel. One connection between the two fields can be found with Cayley graphs and Boolean functions (BF). Accordingly, we can represent a cryptographic Boolean function with a Cayley graph and examine its properties. In this thesis, we convert the substitution boxes within the Data Encryption Standard (DES) to Boolean functions and represent them with Cayley graphs. From the Cayley graph, we analyze the graph spectra and attempt to determine a relationship with the cryptographic properties of the corresponding Boolean functions. With the spectra, we also make some inferences about the structure of the Cayley graph.

14. SUBJECT TERMS

Data Encryption Standard, Boolean Function, Cayley Graph, Graph Spectra

15. NUMBER OF PAGES 179

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

THE SPECTRA OF DES S-BOXES

Mathew B. Fukuzawa
Captain, United States Army
B.S., Michigan State University, 2005

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 2014

Author: Mathew B. Fukuzawa

Approved by: Pantelimon Stănică
Thesis Advisor

Craig Rasmussen
Second Reader

Carlos Borges
Chair, Department of Applied Mathematics

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

We typically do not associate the field of graph theory with the field of cryptography. In graph theory, the aim is to model relationships with a graph and examine properties of that graph. The goal of cryptography is to design a communication system over a nonsecure channel. One connection between the two fields can be found with Cayley graphs and Boolean functions (BF). Accordingly, we can represent a cryptographic Boolean function with a Cayley graph and examine its properties. In this thesis, we convert the substitution boxes within the Data Encryption Standard (DES) to Boolean functions and represent them with Cayley graphs. From the Cayley graph, we analyze the graph spectra and attempt to determine a relationship with the cryptographic properties of the corresponding Boolean functions. With the spectra, we also make some inferences about the structure of the Cayley graph.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

AES Advanced Encryption Standard

ANF algebraic normal form

ATM automated teller machine

BF Boolean function

CBC cipher block chaining

CFB cipher feedback

CTR counter

DES Data Encryption Standard

DFT discrete Fourier transform

ECB electronic codebook

EFF Electronic Frontier Foundation

FIPS Federal Information Processing Standards

GAC global avalanche criteria

IBM International Business Machines

IP initial permutation

IV initialization vector

LFSR linear feedback shift register

NBS National Bureau of Standards

NIST National Institute of Standards and Technology

NPS Naval Postgraduate School

NSA National Security Agency

OFB output feedback

PC propagation criteria

pc personal computer

RSA Rivest-Shamir-Adleman

SAC strict avalanche criteria

S-Box substitution box

SPN substitution-permutation networks

TB terabytes

U.S. United States

WHT Walsh-Hadamard transform

WT Walsh transform

XOR exclusive or

Acknowledgments

First and foremost, I would like to thank God for allowing me to enjoy the many benefits of life. My accomplishments would not be possible without Him.

Second, I thank the United States Army, in particular the United States Military Academy at West Point, for selecting me to attend graduate school and pursue an advanced degree. I know that this opportunity is not afforded to everyone, and I am grateful for the knowledge acquired through my studies.

Third, I would like to thank the faculty of the Applied Mathematics department at the Naval Postgraduate School. Their professionalism, knowledge, and compassion made this an extremely enjoyable experience. In particular, I am grateful for my thesis advisor, Dr. Pantelimon Stănică, who not only spent hours laboring through this thesis, but he is by far one of the most intelligent and brilliant mathematicians I have ever met. I am also thankful for Drs. Ralucca Gera and Craig Rasmussen, whose help in editing my thesis is appreciated and whose love of graph theory helped me to enjoy a deeper understanding of the material. Additionally, I am indebted to Dr. David Canright, whose help in writing Maple code saved me numerous hours in computation. I also thank Lecturer Bard Mansager, who helped integrate me into the program and provided mentorship on countless occassions.

Fourth, I thank my peers in the department for their friendship, support, and guidance. Special thanks go out to Lieutenant Colonels Randy Boucher and Jon Roginski, who provided their wisdom in mentoring me through the maze of professional development.

Last but certainly not least, I thank my family for sticking by my side. My children, Hannah and Luke, are the two accomplishments I am most proud of in life. My wife, Lindsay, is the most loyal person I have ever met. In addition to being my best friend, she is the best mother for our children.

RLTW!

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

Cryptography is often a word that the mainstream population associates with code breaking or secret military intelligence work performed in an underground bunker—this thinking no doubt promoted with movies such as *The Da Vinci Code, Enigma*, and *National Treasure*. While there is perhaps a part of these stereotypes involved, cryptography is much more than this. The mathematics behind cryptography are what keep many of our daily communications secure, i.e., safe enough from prying eyes.

Graph theory is an even more abstract concept for most people. The word *graph* typically generates a mental image so ancient that most people would rather not return to middle school algebra class, where basic functions were plotted on a two-dimensional plane. Graph theory, however, is an emerging field that studies relationships between objects from a mathematical perspective.

1.1 Motivation

The motivation for this thesis came from a desire to connect two prominent areas of discrete mathematics—cryptography and graph theory. The two specific areas linked in this work are the Data Encryption Standard (DES) and spectral graph theory. DES has been analyzed extensively since its inception in the 1970s, mainly in its weaknesses for the purpose of breaking the cipher and improving future algorithms. Some of the prominent researchers of DES include Carlisle Adams, Eli Biham, Ernest Brickell et al., Don Coppersmith, Marc Davio, Martin Hellman, Mitsuru Matsui, Adi Shamir, and Stafford Tavares, just to name a few. On the other hand, spectral graph theory arose circa the same timeframe as the DES, with the intent of deducing properties of a graph from the *spectra* of its associated matrices. By this, we mean that a graph can be represented by a matrix, whose eigenvalues and eigenvectors can be analyzed to determine information about the graph.

Within cryptography, the author was particularly motivated by the works of Claude Carlet, Thomas Cusick, and Pantelimon Stănică, who continue to solidify the role of Boolean functions (BFs) in cryptography. While BFs have their place in logic and circuit design, their

use in cryptography continues to be a topic of relevance. Within spectral graph theory, the classic references are written by Norman Biggs, Dragoš Cvetkovíc et al., and Fan Chung. The more recent work by Stanley Florkowski [1], however, was particularly influential in directing the author's focus to something tangible rather than theoretical.

A BF has a graphical representation, known as a Cayley graph, that can be analyzed in terms of its *spectrum*. The term *spectrum* will become clearer in Chapters 4 and 5, but note that a BF has a representation in terms of a type of spectrum and a graph also has a spectral representation by its eigenvalues. Anna Bernasconi and Bruno Codenotti linked these two spectra with their discovery that a relation exists between the *Walsh spectrum* of a BF and the spectrum of its associated Cayley graph.

Through this point, no one has attempted to analyze the DES in terms of Cayley graph spectra. Some have analyzed the aspects of BFs and their use in block ciphers such as DES, but no one has converted all eight substitution boxes (S-Box) in DES to a set of BFs and analyzed the spectra of their corresponding Cayley graph adjacency matrices.

1.2 Research Questions

DES is a *block cipher* utilizing a substitution step via the aforementioned boxes. These boxes form the nonlinear part of the algorithm and thus contribute to the overall security of the cipher. With this in mind, we aim to explore the following questions:

1. What are the BF representations of the DES S-Boxes?
2. What are the cryptographic properties of these BFs?
3. What properties of the associated Cayley graphs can be deduced from spectral graph theoretic techniques?
4. Is there a relationship between the Cayley graph spectra and the cryptographic properties of the associated BFs?
5. Do the DES S-Box BFs satisfy the propagation criteria (PC) of degree k?

1.3 Thesis Organization

Through the process of investigating the research questions, this thesis is organized in the following manner:

- Chapter 2 discusses the necessary background in algebra and number theory.
- Chapter 3 reviews basic concepts of cryptography and also discusses the organization of DES.
- Chapter 4 discusses BFs and their application in cryptography.
- Chapter 5 reviews graph theory terminology and introduces spectral graph theory.
- Chapter 6 examines the DES S-Boxes as BFs and their associated Cayley graphs.
- Chapter 7 extends the notion of propagation criteria to the DES BFs.
- Chapter 8 summarizes the results of this thesis and includes areas for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Preliminaries on Algebra and Number Theory

This introductory chapter and the several that follow establish the foundation upon which the mathematics presented in this thesis depend. The algebra presented here goes beyond our usual idea of arithmetic, in that we consider familiar operations and sets on an abstract level. This chapter is by no means all-inclusive and the interested reader should consult some of the more classic texts on abstract algebra by John Fraleigh [2] and Thomas Hungerford [3].

2.1 Number Theory

Number theory is primarily the study of the set of integers and their properties [4]. These topics essentially bridge the gap between basic arithmetic and advanced algebra. The definitions presented in this section are taken from [3].

2.1.1 Divisibility

A set is an unordered collection of objects. We assume that the reader is familiar with some basic mathematical sets of numbers as follows:

$$\mathbb{N} = \{0, 1, 2, 3, \ldots, \}$$
$$\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots, \}$$
$$\mathbb{Q} = \left\{ \frac{p}{q} : p, q \in \mathbb{Z}, q \neq 0 \right\}.$$

Definition 2.1.1. Let $a, b \in \mathbb{Z}$ with $a \neq 0$. Then a **divides** b, or a is a **divisor** of b, or b is a **multiple** of a if $b = ak$ for some integer k. We denote this by $a|b$.

Definition 2.1.2. A nonzero integer p is called **prime** if its only divisors are ± 1 and $\pm p$.

EXAMPLE 2.1.3. -5, 3, 11, and 29 are prime but 24 is not.

Definition 2.1.4. Let $a, b \in \mathbb{Z}$, not both zero. The **greatest common divisor (gcd)** of a and b is the largest $d \in \mathbb{Z}$ that divides both a and b. Equivalently, d is the *gcd* of a and b

provided that:

(*i*) $d|a$ and $d|b$;
(*ii*) $c|a$ and $c|b \implies c \leq d$. (for all $c \in \mathbb{Z}^+$)

EXAMPLE 2.1.5. The *gcd* of 8 and 36 is 4.

Definition 2.1.6. If $gcd(a,b) = 1$, then a and b are called **relatively prime**.

EXAMPLE 2.1.7. 9 and 25 are *relatively prime*.

2.1.2 Congruence

This section continues the concept of divisibility, while also introducing congruence and congruence classes. Once again, these definitions and concepts are taken from [3].

Definition 2.1.8. Let $a,b,n \in \mathbb{Z}$ with $n > 0$. Then ***a* is congruent to *b modulo n*** provided that $n|(a-b)$ or $n|(b-a)$. Note: This is written as $a \equiv b \pmod{n}$.

EXAMPLE 2.1.9. $23 \equiv 11 \pmod{6}$ since $6|(23-11)$. Also, $4 \equiv 13 \pmod{3}$ since $3|(13-4)$.

If we alter the second part of Example 2.1.9, note that $4 \equiv 16 \pmod{3}, 4 \equiv 19 \pmod{3}, 4 \equiv 22 \pmod{3}, \ldots$ This allows us to define the notion of a congruence class.

Definition 2.1.10. Let $a,n \in \mathbb{Z}$ with $n > 0$. The **congruence class of *a modulo n*** (denoted $[a]$) is the set of all integers congruent to *a modulo n*, i.e.,

$$[a] = \{b : b \in \mathbb{Z} \quad \text{and} \quad b \equiv a \pmod{n}\}.$$

EXAMPLE 2.1.11. In congruence modulo 4, $[3] = \{\ldots, -9, -5, -1, 3, 7, 11, 15, 19, \ldots\}$, sometimes also denoted $[3]_4$. Also, note that $[3]_4 = [-1]_4$. In some circles, $[3]_4$ is also called the *residue class* of 3 mod 4.

The next logical question is how many congruence classes are there for a given n? After all, $[3]_4 = [-1]_4 = [7]_4 = [11]_4 = \cdots$, but $[2]_4 \neq [3]_4$. The answer lies in Definition 2.1.12.

Definition 2.1.12. The set of all congruence classes *modulo n* is a partitioning of the set $\mathbb{Z}$ into n distinct equivalence classes, given by

$$\mathbb{Z}_n = \{[0],[1],[2],\ldots,[n-1]\}.$$

EXAMPLE 2.1.13. $\mathbb{Z}_4 = \{[0],[1],[2],[3]\}$. This means that the elements of $\mathbb{Z}_4$ are congruence classes and not integers. Here are the elements of $\mathbb{Z}_4$:

$$[0] = \{\ldots,-8,-4,0,4,8,12,\ldots\}$$
$$[1] = \{\ldots,-7,-3,1,5,9,13,\ldots\}$$
$$[2] = \{\ldots,-6,-2,2,6,10,14,\ldots\}$$
$$[3] = \{\ldots,-5,-1,3,7,11,15,\ldots\}.$$

The important distinction here is that while each congruence class in $\mathbb{Z}_n$ has infinitely many elements [3], there are only a finite number of distinct congruence classes in $\mathbb{Z}_n$. Thus, while it is true that $[-3]_4 = [1]_4 = [5]_4 = [9]_4$, the distinct classes of $\mathbb{Z}_4$ are $[0],[1],[2],\text{and}[3]$.

2.1.3 Modular Arithmetic

Ever since grade school, we have performed operations on the integers. The integers, however, are an infinite set, and the set we are interested in, $\mathbb{Z}_n$, is a finite set. We would like a way to perform operations on $\mathbb{Z}_n$, and this is where modular arithmetic emerges.

Returning to the idea of congruence, recall that $a \equiv b \pmod{n} \iff n|(a-b)$. This number n is called the *modulus*, and in the context of this congruence, *mod* represents a relation on the integers [4]. We now introduce some new notation that is closely related.

If we were asked to compute $\frac{11}{4}$ in grade school, most of us resorted to long division

$$\begin{array}{r} 2. \\ 4\overline{)\,11} \\ \underline{8} \\ 3 \end{array}$$

In traditional grade school terminology, 4 is the *divisor*, 11 is the *dividend*, 2 is the *quotient*, and 3 is the *remainder*. In the context of abstract algebra and cryptography, the remainder (sometimes called the *residue*) is often the object that garners the most attention.

Definition 2.1.14. The notation $r = a \textbf{ mod } d$, where a is the dividend, d is the divisor, and r is the remainder, represents the smallest positive remainder when a is divided by d.

EXAMPLE 2.1.15. $11 \textbf{ mod } 4 = 3$, $-7 \textbf{ mod } 4 = 1$, $7 \textbf{ mod } 4 = 3$, $136 \textbf{ mod } 13 = 6$. Note: $-7 \textbf{ mod } 4 = 1$ since $-7 = 4(-2) + 1$ as a result of the division algorithm (omitted by assumption of reader knowledge).

The notation $\textbf{mod}\, n$ is a function, but is closely related to the *mod* defined in congruence. The relationship is given by Theorem 2.1.16 [4].

Theorem 2.1.16. Let $a, b \in \mathbb{Z}$ and let $n \in \mathbb{Z}^+$ (set of positive integers). Then $a \equiv b \pmod{n} \iff a \textbf{ mod } n = b \textbf{ mod } n$.

Proof: $(\rightarrow)$ $a \equiv b \pmod{n} \implies n|(a-b) \implies a - b = nk, k \in \mathbb{Z}(*)$.
Then $a = nk + b$, so we let $r = a \textbf{ mod } n$. Then $\exists q \in \mathbb{Z}$ such that $a = nq + r, 0 \leq r < n$ by the Division Algorithm. Now substitute $a = nq + r$ into (*).

$$\begin{aligned} nq + r - b &= nk \\ n(q-k) + r &= b, \quad (q-k) \in \mathbb{Z} \\ \implies r &= b \textbf{ mod } n \\ \therefore b \textbf{ mod } n &= a \textbf{ mod } n \end{aligned}$$

$(\leftarrow)$ Let $r = a \textbf{ mod } n = b \textbf{ mod } n$. Then $a = nq_1 + r$ and $b = nq_2 + r$. Solving these equations for r, we have $r = a - nq_1$ and $r = b - nq_2$. Therefore,

$$\begin{aligned} a - nq_1 &= b - nq_2 \\ a - b &= nq_1 - nq_2 \\ a - b &= n(q_1 - q_2), \quad (q_1 - q_2) \in \mathbb{Z} \\ &\implies n|(a-b) \\ &\implies a \equiv b \pmod{n} \end{aligned}$$

Armed with this knowledge, we can now define arithmetic on $\mathbb{Z}_n$. The two operations that we are concerned with are addition and multiplication.

Definition 2.1.17. Addition and multiplication in $\mathbb{Z}_n$ are defined by

$$[a]_n + [b]_n = [a+b]_n = (a+b) \textbf{ mod } n$$
$$[a]_n \cdot [b]_n = [ab]_n = (a \cdot b) \textbf{ mod } n.$$

EXAMPLE 2.1.18. In $\mathbb{Z}_6$, $[3]+[2]=[5]$, $[4]+[5]=[3]$, and $[3]\cdot[2]=[0]$.

2.2 Abstract Algebra Concepts

The remaining portion of this chapter will focus on the abstract algebra concepts at the heart of cryptography. For a truly deep understanding of these topics, the reader should consult an algebra reference with a cryptographic focus such as Fraleigh [2] or Rudolf Lidl and Harald Niederreiter [5].

2.2.1 Binary Operations

We first need to define a few operations on mathematical sets. It is assumed that the reader has some basic knowledge of set theory.

Definition 2.2.1. Let A and B be sets. The **Cartesian product** of A and B is given by the set $A \times B$, defined [2] as

$$A \times B = \{(a,b) : a \in A \text{ and } b \in B\}.$$

EXAMPLE 2.2.2. If $A = \{a,b\}$ and $B = \{y,z\}$, then $A \times B = \{(a,y),(a,z),(b,y),(b,z)\}$.

For the purposes of upcoming material, we will often be concerned with the Cartesian product of two sets which are the same, i.e., $A \times A$. Consider $\mathbb{Z}$, the set of integers, and the familiar operation of addition. If we take two arbitrary integers, say u and v, and perform addition on them, we get back another integer w (of course w may or may not be equal to u or v). We have just defined, albeit informally, a binary operation on $\mathbb{Z}$.

Definition 2.2.3. A **binary operation** [2] on a nonempty set S is a function mapping $S \times S$ into S, given mathematically as $f : S \times S \to S$.

This operation is symbolized by $*$, to indicate any general function satisfying the definition. For example, addition is not the only binary operation on $\mathbb{Z}$ (multiplication as well). In other words, assuming $(a,b) \in S \times S$, a binary operation $*$ on S assigns (a,b) to $a*b \in S$.

2.2.2 Groups

We now turn our attention to one of the oldest algebraic systems in mathematics—*groups*. Group theory, or the study of groups, was introduced by Évariste Galois. In this sense, group theory is also known as Galois theory. Galois was a 19th century French mathematician who lived just 20 years, meeting his fate following a pistol duel. Despite spending the majority of his teen years trying to gain acceptance into school and failing, Galois did manage to record his discoveries. One of these results involved the solvability of an algebraic equation of high order using radicals; the method became known as group theory [6].

Definition 2.2.4. A **group** is a nonempty set G together with a binary operation $*$ that satisfies the following axioms:

1. *Closure*: If $a,b \in G$, then $a*b \in G$.[1]
2. *Associativity*: $(a*b)*c = a*(b*c) \quad \forall\, a,b,c \in G$.
3. Existence of an *identity*: $\exists\, e \in G$ such that $\forall\, a \in G, \quad a*e = a = e*a$.
4. Existence of an *inverse*: $\forall\, a \in G$, $\exists\, a' \in G$ such that $\quad a*a' = a'*a = e$.

   A group is **abelian** (sometimes called commutative) if it also satisfies the following axiom:
5. *Commutativity*: $a*b = b*a \quad \forall\, a,b \in G$.

EXAMPLE 2.2.5. $\langle \mathbb{Z}, + \rangle$ is an abelian group. The sum of any two integers is another integer; the addition is associative. The identity element in $\mathbb{Z}$ is 0 and the inverse element is just the element of opposite sign. Also, addition of integers is commutative.

EXAMPLE 2.2.6. $\langle \{[0],[1],[2],\ldots,[n-1]\}, [a+b]_n \rangle$ is a group under addition *modulo n*.

[1] Some texts do not include this axiom since closure is an inherent property of a binary operation.

EXAMPLE 2.2.7. The set of all $n \times n$ matrices with real entries under matrix multiplication is *not* a group. In particular, the zero matrix has no inverse.

With regard to Examples 2.2.5 and 2.2.6, $\langle \mathbb{Z}, + \rangle$ is an example of an *infinite* group because it contains infinitely many elements. The second example is a *finite* group because it contains a finite number of elements. The number of elements in a finite group G is the **order** of the group [5]. For those familiar with set theory, this term is analogous to the *cardinality* of a finite set. We also sometimes refer to a group under addition as an *additive group*, while a group whose binary operation is multiplication is called a *multiplicative group*.

A convenient way to display a group under its binary operation is via the Cayley table, sometimes also called a group table or addition/multiplication table. In this table, the elements of a group G are placed along the top row and leftmost column, and the (i, j) entry in this table represents $a_i * b_j$. For example, Table 2.1 displays the group $\mathbb{Z}_5$ under addition *modulo* 5.

+	[0]	[1]	[2]	[3]	[4]
[0]	[0]	[1]	[2]	[3]	[4]
[1]	[1]	[2]	[3]	[4]	[0]
[2]	[2]	[3]	[4]	[0]	[1]
[3]	[3]	[4]	[0]	[1]	[2]
[4]	[4]	[0]	[1]	[2]	[3]

Table 2.1: The Cayley Table for $\mathbb{Z}_5$.

There is much more detail in the realm of group theory, but that is beyond the knowledge required for this thesis. The interested reader should consult [2, 5] for a deeper look.

2.2.3 Rings

We now move on to the concept of a *ring*, in which two binary operations and additional axioms are now defined. While the origins of a ring date back to the mid-19th century, the formal definitions of a ring and ring theory did not appear until the early 1900s. William R. Hamilton first described a complex number system coined the *quaternions*, in which he attempted to apply vector algebra to 3-dimensional space. This formed the basis upon which

subsequent mathematicians attempted to study finite commutative and noncommutative algebras. Israeli mathematician Abraham Fraenkel and Japanese Shezo Sono are credited with defining the concept of a ring in 1914 and 1917, respectively. Emmy Noether and Emil Artin formally theorized rings in the 1920s, and ring theory took off from there with the works of Wolfgang Krull and others [6, 7].

Definition 2.2.8. A **ring** $\langle R, +, \cdot \rangle$ is a nonempty set R together with two binary operations $+$ and $\cdot$, which we call *addition* and *multiplication*, such that the following axioms are satisfied [2, 5]:

1. $\langle R, + \rangle$ is an abelian group.
2. Multiplication is associative, i.e., $(a \cdot b) \cdot c = a \cdot (b \cdot c) \quad \forall\, a, b, c \in R$.
3. The distributive laws hold, i.e., $\forall\, a, b, c, \in R$, we have $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(b + c) \cdot a = b \cdot a + c \cdot a$.

EXAMPLE 2.2.9. The set of integers $\mathbb{Z}$ is a ring with the usual addition and multiplication. Verification of the axioms is left to the reader.

Some rings have additional special properties that are worth noting. A ring is *commutative* if the multiplication operation $\cdot$ is commutative. Also, a ring is called a *ring with identity* if R contains a multiplicative identity, i.e., there exists an element e such that $a \cdot e = a = e \cdot a \quad \forall\, a \in R$. Thus, $\mathbb{Z}$ is a commutative ring with identity [5].

EXAMPLE 2.2.10. The set of even integers with the usual operations is a ring; in fact it is a commutative ring. The set of odd integers is not a ring since closure under addition is not satisfied.

EXAMPLE 2.2.11. The sets $\mathbb{Q}, \mathbb{C}$, and $\mathbb{R}$ are all commutative rings with identity.

2.2.4 Fields

The interesting thing about fields is that mathematicians were studying them well before the formal concept of a ring was defined, yet we often define fields as a special type of

ring. Niels Abel and Galois inferred the idea of a *field* with their work on the solvability of equations circa the 1830s; it was not until 1879, when Richard Dedekind published an explicit definition for a field, that stimulation in the subject arose. Dedekind focused on infinite sets, whereas Heinrich Weber discussed the notion of finite fields in 1893. It was Galois, however, that perhaps influenced the development of field theory the most. As a result, finite fields are also known as Galois fields [2, 6].

Definition and Examples

Definition 2.2.12. A **field** F is a commutative ring R with identity $e \neq 0$ also satisfying the following axiom [3]:

- $\star$ $\forall\, a \neq 0 \in R$, the equation $ax = e$ has a solution in R [every nonzero element has a multiplicative inverse].

An alternative definition of a field given by Fraleigh [2] and Lidl and Niederreiter [5] is perhaps more appealing to the mathematically inclined:

Definition 2.2.13.
(i) A ring with a multiplicative identity is called a *ring with identity*; the identity is often called *unity*.
(ii) A ring in which multiplication is commutative is called a *commutative ring*.
(iii) A ring is an *integral domain* if it is a commutative ring with identity $e \neq 0$ in which $ab = 0 \implies a = 0$ or $b = 0$.
(iv) A ring is called a *division ring* if the nonzero elements of R form a group under multiplication (every nonzero element has a multiplicative inverse in R).
(v) A commutative division ring is called a **field**.

Breaking down Definition 2.2.13, a field is a ring on which two binary operations (called multiplication and addition) are defined, also containing a unique zero element and identity $e \neq 0$. Since a field is a commutative division ring, its nonzero elements form an abelian

group under multiplication. Part (iii) of the definition guarantees that a field has no zero divisors, since all nonzero elements have a multiplicative inverse.

EXAMPLE 2.2.14. $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{C}$ are all fields. However, $\mathbb{Z}$ is not a field since not all nonzero elements have a multiplicative inverse, e.g., $3x = 1$ has no solution in $\mathbb{Z}$.

EXAMPLE 2.2.15. In general, $\mathbb{Z}_n$ is not an integral domain and thus not a field, but when $n = p$ a prime, $\mathbb{Z}_p$ is an integral domain and thus a field (proof omitted). For example, in $\mathbb{Z}_4$ we have $2 \cdot 2 = 0$ but $2 \neq 0$.

Finite Fields

Example 2.2.15 from above illustrates a concept which is at the heart of cryptography, that of the finite field. A *finite field* is a field that contains only finitely many elements. While the theory of finite fields is very deep and mathematical, the background presented here is enough to give the reader a baseline of knowledge. Lidl & Niederreiter [5] devote an entire text to the subject.

Recall that we denoted the set of all congruence classes *modulo n* as $\mathbb{Z}_n$. By noting that this set is also the set of possible remainders when a positive integer is divided by n, we can also refer to this as the set of residue classes $\mathbf{mod}\, n$. We now define an *ideal*, which is a subring J of a ring R such that for all $a \in J$ and $r \in R$ we have $ar \in J$ and $ra \in J$. Note, for J to be a subring, J must be closed under $+$ and $\cdot$ and also satisfy the ring axioms. An ideal J partitions a ring R into disjoint sets (called *cosets*); these disjoint sets are the *residue classes modulo J*. The entire set of residue classes *modulo J* form a ring with the operations induced from the operations of R (proof omitted), called the *residue class ring* of R *modulo J*, symbolized by R/J [5]. Depending on the source, some texts also call this R/J the *factor ring* (or *quotient ring*) of R by J [2].

When we consider our example from above, $\mathbb{Z}_n$, the residue class ring $\mathbb{Z}/(n)$ contains the following elements:

$$[0] = 0 + (n), [1] = 1 + (n), \ldots, [n-1] = n - 1 + (n).$$

Instead of (n), some texts also use the notation $n\mathbb{Z}$ to represent the ideal of $\mathbb{Z}$ in the factor ring $\mathbb{Z}/n\mathbb{Z}$. The notation (n) is the same as $n\mathbb{Z}$; it is the principal ideal generated by n,

i.e., the set of all multiples of n in $\mathbb{Z}$. While not shown here, $\mathbb{Z}_n$ is isomorphic to $\mathbb{Z}/n\mathbb{Z}$, i.e., there is an injective and surjective homomorphism between the two (preserving the respective operations). Since $\mathbb{Z}_n$ is a field if and only if $n = p$ a prime, then the factor ring $\mathbb{Z}/n\mathbb{Z}$ is a field if and only if n is a prime [2].

The residue class fields $\mathbb{Z}/(p)$ where p is a prime form the basis for the finite fields used in this thesis. We would like a more convenient representation and usage of these residue class fields. A mapping is a convenient way to transfer the structure from one set to another [5]. The set without structure will be denoted by $GF(p) = \{0, 1, \ldots, p-1\}$, where this is a set of integers with p elements. Let $\phi : \mathbb{Z}/(p) \to GF(p)$ be a bijective mapping defined by $\phi([a]) = a$ for $a = 0, 1, \ldots, p-1$. It is not too difficult to show that ϕ is also a homomorphism, i.e., $\phi([a] + [b]) = \phi([a]) + \phi([b])$ and $\phi([a][b]) = \phi([a])\phi([b])$. Since this mapping is a bijective homomorphism, it can also be called an isomorphism, whereby the structure on $GF(p)$ is induced by ϕ. Moreover, since $\mathbb{Z}/(p)$ is a field when p is prime, then $GF(p)$ is a field induced by ϕ. Note, we are not stating that the elements of $\mathbb{Z}/(p)$ and $GF(p)$ are the same, only that the structure of a finite field is transferred between the two.

The finite field $GF(p)$ is so important that it is called the *Galois field of order p* after É. Galois. For conciseness, Galois fields are also denoted by $\mathbb{F}_p$ and will henceforth be referred to as such in this thesis. Since the elements of $\mathbb{F}_p$ are ordinary integers, arithmetic in the field is carried out *modulo p*.

Consider the following example for $\mathbb{F}_2 = \{0, 1\}$ in Table 2.2.

+	0	1
0	0	1
1	1	0

·	0	1
0	0	0
1	0	1

Table 2.2: The Addition and Multiplication Tables for $\mathbb{F}_2$, after [5].

There are a few more things to say about Galois fields, but first we need a short buildup. We define the *characteristic* of a ring as the least positive integer n such that $na = 0$ for all elements a in the ring (if such n exists, otherwise the ring has characteristic 0) [2]. For example, the ring $\mathbb{Z}_n$ has characteristic n and the ring $\mathbb{Q}$ has characteristic 0.

Let F be a field and K a subfield of F (a subset that is also a field and closed under the usual operations). Then F is an *extension field* of K [5]. Now, if E is an extension field of F with dimension n as a vector space over F (see next subsection), then E is a *finite extension of degree n over F*. If a finite field F has q elements, then E has q^n elements assuming E is a finite extension of degree n over F. We can also regard E as a vector space (see next section) of dimension n over F.

If E is a finite field of characteristic p a prime, then E contains exactly p^n elements for some positive integer n [2]. This result follows from the previous paragraph. This result implies for every prime p and every positive integer n, there exists exactly one finite field with p^n elements, i.e., $GF(p^n) = \mathbb{F}_{p^n}$ exists, and moreover, it is unique up to an isomorphism.

Polynomials

When we think of our usual idea of a polynomial, we remember something like x^2+2x+1 from high school. In general, a polynomial can be written as $a_0+a_1x+\cdots+a_nx^n$, or as a sum $\sum_{i=0}^{n} a_ix^i$. We now expand this concept to rings.

Let R be a ring. A polynomial over R is an expression of the form

$$f(x) = \sum_{i=0}^{n} a_ix^i = a_0 + a_1x + \cdots + a_nx^n, \tag{2.1}$$

where n is a nonnegative integer and the a_i are elements of R [5]. The symbol x is no longer called a *variable*, but rather an *indeterminate*; x does not belong to R. Since R is a ring, we also need to define its two binary operations, addition and multiplication. Let $f(x) = \sum_{i=0}^{n} a_ix^i$ and $g(x) = \sum_{i=0}^{n} b_ix^i$. The sum of $f(x)$ and $g(x)$ is given by $f(x)+g(x) = \sum_{i=0}^{n} (a_i+b_i)x^i$. Now, let $f(x) = \sum_{i=0}^{n} a_ix^i$ and $g(x) = \sum_{j=0}^{m} b_jx^j$. Then the product is given by $f(x)g(x) = \sum_{k=0}^{n+m} c_kx^k$, where $c_k = \sum_{i+j=k} a_ib_j$.

This ring R together with the addition and multiplication operations above is called the *polynomial ring* over R [5] and is denoted by $R[x]$.

EXAMPLE 2.2.16. In $\mathbb{F}_2[x]$, the expansion of $(x+1)^2$ is $(x+1)^2 = (x+1)(x+1) = x^2+2x+1 = x^2+1$ and in general, $(x+a)^{2^n} = x^{2^n} + a^{2^n}$.

For cryptography purposes, we are more interested in polynomials over fields but the approach is somewhat different. Let F be a field. Then $F[x]$ is an integral domain but not a field since x does not have a multiplicative inverse in $F[x]$, i.e., in $F[x]$, $xf(x) = 1$ has no solutions [2]. We can get around the fact that $F[x]$ is not a field because every integral domain has a field of quotients. This field of quotients is denoted by $F(x)$ and consists of all quotients of the form $f(x)/g(x)$, with $f(x)$ and $g(x)$ polynomials in $F[x]$ and $g(x) \neq 0$ [2]. $F(x)$ is also called the field of *rational functions* over F; its elements are called *rational functions*.

EXAMPLE 2.2.17. In the most general sense, $F[x]$ is the ring of polynomials with coefficients in some arbitary field F. $\mathbb{F}_5[x]$ consists of all polynomials whose coefficients are in $\mathbb{F}_5$.

We now proceed to develop and define two more concepts which are essential to cryptographic functions–*irreducibility* and *primitivity*. Since $F[x]$ has a field of quotients, it is natural to expect operations such as division and factoring are present. In fact, they are and just like with integers, the division algorithm can be applied to polynomials in $F[x]$. Likewise, a *greatest common divisor* also exists in $F[x]$ as well as a *least common multiple*. The notion of a *prime* polynomial also exists and the concept is analogous to the integers. Two polynomials f and g are *relatively prime* if $gcd(f,g) = 1$. Similarly, a polynomial $p(x)$ is *prime* if it has the property that it divides the product $f(x)g(x)$ only when it divides one of $f(x)$ or $g(x)$. In other words, the only factors of $p(x)$ have either the same degree as p or degree zero.

EXAMPLE 2.2.18. $p(x) = x^2 + 1$ is prime in $\mathbb{R}[x]$ since it does not factor into a product $f(x)g(x)$, where $f(x)$ and $g(x)$ are polynomials with real coefficients. It is, however, not prime (i.e., *composite*) in $\mathbb{C}[x]$!

Definition 2.2.19. A polynomial $p \in F[x]$ is **irreducible over F** (or *irreducible in* $F[x]$, or *prime in* $F[x]$) if p has positive degree and $p = bc$ with $b, c \in F[x]$ implies that either b or c is a constant polynomial.

In other words, an irreducible polynomial cannot be factored further except for a trivial factorization, i.e., p cannot be expressed as a product gh both of lower degree than the

degree of p [2, 5]. It should be apparent that the prime elements of $F[x]$ are the irreducible polynomials over a field F. Example 2.2.20 illustrates the idea of irreducibility.

EXAMPLE 2.2.20. $x^2-2 \in \mathbb{Q}[x]$ is irreducible over the field $\mathbb{Q}$ of rationals since it has no zeros in $\mathbb{Q}$. However, x^2-2 is *reducible* over $\mathbb{R}$ since it factors in $\mathbb{R}[x]$ into $(x+\sqrt{2})(x-\sqrt{2})$.

With the notion of an irreducible polynomial, we can now develop the idea of *primitive* polynomials. First we need to define the *order* of a nonzero polynomial over a finite field, taken from Lidl et al.

Definition 2.2.21. Let $f \in \mathbb{F}_q[x]$ be a nonzero polynomial. If $f(0) \neq 0$, then the least positive integer e for which $f(x)|(x^e-1)$ is called the **order** of f and denoted by ord(f) or ord($f(x)$).

EXAMPLE 2.2.22. Let $f(x) = x^4+x^3+1$ be a polynomial in $\mathbb{F}_2[x]$. The *order* of f is 15, since $(x^4+x^3+1)|(x^{15}-1)$. Note that, since we are in $\mathbb{F}_2$, subtraction performs the same as addition and we may perform long division to check divisibility.

$$(x^4+x^3+1)|(x^{15}-1) \implies \frac{x^{15}+1}{x^4+x^3+1} = x^{11}+x^{10}+x^9+x^8+x^6+x^4+x^3+1$$

Now we can present the notion of a *primitive* polynomial. Primitive polynomials are used in multiple cryptographic applications, such as generating maximal-period linear feedback shift registers (LFSRs) or pseudorandom numbers. Primitive polynomials are also used in many well-known algorithms such as Advanced Encryption Standard (AES).

Definition 2.2.23. A polynomial $f \in \mathbb{F}_q[x]$ of degree m is a **primitive** polynomial over the field $\mathbb{F}_q$ if f is monic, $f(0) \neq 0$, and ord(f) $= q^m-1$.

Note that in Definition 2.2.23 [5], the term *monic* means that the coefficient of the highest degree term is one. A primitive polynomial is a monic, irreducible polynomial over $\mathbb{F}_q$ and has a root $\alpha \in \mathbb{F}_{q^m}$ that generates the entire multiplicative group of $\mathbb{F}_{q^m}$. This is why many applications such as AES use primitive polynomials —they generate the entire Galois field used in the algorithm. Although it is true that a primitive polynomial is irreducible, it is not always true that an irreducible is primitive.

EXAMPLE 2.2.24. The polynomial in Example 2.2.22 is irreducible and primitive. As a check, f is monic since the coefficient of x^4 is one. We now check that 0 is not a *zero* (aka root) of the polynomial, and we see that $f(0) = 0+0+1 = 1$. Finally, we require that $\text{ord}(f) = 2^4 - 1$, which was verified as 15 previously.

2.2.5 Vector Spaces

Most readers are familiar with the concept of a vector space from a typical course in linear algebra. In a common text such as Steven Leon [8], a vector space is defined with the natural Euclidean approach. A vector space has two defined operations: addition and scalar multiplication, whereby these operations can be performed on any vector within the vector space. Consider the familiar two-dimensional world, or $x-y$ plane denoted by $\mathbb{R}^2$. Any two vectors in $\mathbb{R}^2$ can be added together to produce another vector in $\mathbb{R}^2$; any vector in $\mathbb{R}^2$ can be multiplied by a scalar in $\mathbb{R}$ to also yield another vector in $\mathbb{R}^2$. This is just one example of a vector space in which closure of addition and scalar multiplication is satisfied. Formally, Leon defines a vector space in the following manner.

Definition 2.2.25. Let V be a set on which the operations of addition and scalar multiplication are defined. By this we mean that, with each pair of elements $\mathbf{x}$ and $\mathbf{y}$ in V, we can associate a unique element $\mathbf{x}+\mathbf{y}$ that is also in V, and with each element $\mathbf{x}$ in V and each scalar $\alpha \in \mathbb{R}$, we can associate a unique element $\alpha\mathbf{x}$ in V. The set V, together with the operations of addition and scalar multiplication, is said to form a **vector space** if the following axioms are satisfied:

A1. $\mathbf{x}+\mathbf{y} = \mathbf{y}+\mathbf{x}$ for any $\mathbf{x}$ and $\mathbf{y}$ in V.
A2. $(\mathbf{x}+\mathbf{y})+\mathbf{z} = \mathbf{x}+(\mathbf{y}+\mathbf{z})$ for any $\mathbf{x},\mathbf{y}$ and $\mathbf{z}$ in V.
A3. There exists an element $\mathbf{0}$ in V such that $\mathbf{x}+\mathbf{0} = \mathbf{x}$ for each $\mathbf{x} \in V$.
A4. For each $\mathbf{x} \in V$, there exists an element $-\mathbf{x}$ in V such that $\mathbf{x}+(-\mathbf{x}) = \mathbf{0}$.
A5. $\alpha(\mathbf{x}+\mathbf{y}) = \alpha\mathbf{x}+\alpha\mathbf{y}$ for each scalar α and any $\mathbf{x}$ and $\mathbf{y}$ in V.
A6. $(\alpha+\beta)\mathbf{x} = \alpha\mathbf{x}+\beta\mathbf{x}$ for any scalars α and β and any $\mathbf{x} \in V$.
A7. $(\alpha\beta)\mathbf{x} = \alpha(\beta\mathbf{x})$ for any scalars α and β and any $\mathbf{x} \in V$.
A8. $1 \cdot \mathbf{x} = \mathbf{x}$ for all $\mathbf{x} \in V$.

This is a fine definition for purposes of linear algebra, but it can be generalized using the concepts of groups and fields.

Definition 2.2.26. [2] Let F be a field. A **vector space over** F is an additive abelian group V together with a scalar multiplication of each element of V by each element of F on the left, such that for all $a,b \in F$ and $\alpha,\beta \in V$, the following conditions are satisfied:

$\mathscr{V}_1$. $a\alpha \in V$.
$\mathscr{V}_2$. $a(b\alpha) = (ab)\alpha$.
$\mathscr{V}_3$. $(a+b)\alpha = (a\alpha)+(b\alpha)$.
$\mathscr{V}_4$. $a(\alpha+\beta) = (a\alpha)+(a\beta)$.
$\mathscr{V}_5$. $1\alpha = \alpha$.

In Definition 2.2.26, the elements a,b of an arbitrary field F are *scalars*, while α,β are *vectors*.

EXAMPLE 2.2.27. The additive abelian group of all 2×2 matrices over the reals with the usual scalar multiplication involving matrices is a vector space over $\mathbb{R}$.

EXAMPLE 2.2.28. The complex numbers $\mathbb{C}$ form a vector space over the real numbers.

The *dimension* of a vector space V is the number of linearly independent vectors needed to *span* or *generate* V. With this in mind, the dimension of $\mathbb{R}^2$ is two. A more applicable example to Definition 2.2.26 follows.

EXAMPLE 2.2.29. Let F be a field with E an extension field of F. Also, let $\alpha \in E$, where α is an *algebraic* over F. By *algebraic*, we mean that there exists a non-zero polynomial $f(x) \in F[x]$ such that $f(\alpha) = 0$. Now suppose that the degree of α over F is n. Then we can express the vectors in $F(\alpha)$ as a linear combination such that $\{1,\alpha,\alpha^2,\ldots,\alpha^{n-1}\}$ are linearly independent in $F(\alpha)$ over F. This set of vectors also spans $F(\alpha)$, and thus it has dimension n.

CHAPTER 3:
Block Ciphers

This chapter introduces cryptography and the necessary information on block ciphers. In particular, an overview of the DES is presented with an eye towards each S-Box within the algorithm. For more information on block ciphers and other symmetric algorithms, the reader should refer to [9–11].

3.1 Introduction

Cryptography is the process of designing communication systems over nonsecure channels. The word *cryptography* is often used interchangeably with *cryptology*, though the latter is technically the general word for the study of communication over nonsecure channels [12]. Historically, we might say that the origins of cryptography date back to primitive man and his method of communication with others. The first true example of cryptography, however, probably lies with the ancient Egyptians and their use of hieroglyphics. No matter the civilization nor the timeline, the need to protect information has always been present. The latter half of the 20th century introduced the digital computer, which ultimately made cryptography a required part of everyday life. Unfortunately, as technology advances, so do the means by which adversaries break these systems (known as *cryptanalysis*). As stated in [11]: "Cryptography is the only practical means for protecting the confidentiality of information transmitted through potentially hostile environments, where it is either impossible or impractical to protect the information by conventional physical means."

3.2 Secure Communications

The need for cryptographic algorithms to protect data arises from the basic communication scenario between two (or multiple) people or entities. Cryptography introduces an algorithm or method to convert a message into an encrypted message and vice versa, so that two parties can communicate securely and not have their message read by another party.

3.2.1 Background

Consider the following scenario referenced in Figure 3.1. In this classic figure, two parties, Alice and Bob, want to communicate with each other. Meanwhile, a potential adversary named Eve (Eve for *eavesdropper*), wants to intercept this message.

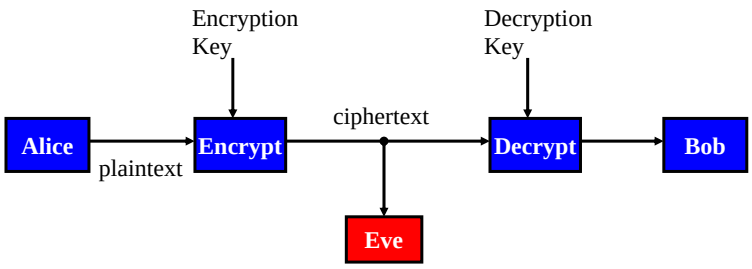


Figure 3.1: The Basic Communication Scenario for Cryptography, after [12].

Alice could send Bob a message in the clear, i.e., unencrypted, but Eve could easily intercept it. Instead, Alice creates a *plaintext* message and encrypts it using an encryption *key*. Once encrypted, the message is now referred to as *ciphertext*. Bob receives the ciphertext and decrypts it back to plaintext using a decryption key. Keeping the contents of the message secure from Eve not only depends on the encryption/decryption method used, but more so on the keys. Encryption and decryption are encompassed in a *cipher*.

The algorithm and the keys together comprise a cryptosystem. With the exception of the one-time pad[2], every cryptosystem can theoretically be broken. Thus, great care is taken to create a cryptosystem that is mathematically too difficult to break in any reasonable amount of time. Claude Shannon introduced the concepts of *confusion* and *diffusion* in regards to good cryptosystem design. Confusion means that it is too difficult for an adversary to detect the outcome of the ciphertext from a one character change in the plaintext. In an algorithm with good confusion, the relationship between the plaintext/key and the ciphertext is often complex. On the other hand, diffusion means that few changes in the plaintext create many changes in the ciphertext. Thus, good diffusion implies that Eve needs a large portion of ciphertext to determine the algorithm and conduct a statistical attack [13].

[2]In a one-time pad, the plaintext is encrypted one character at a time with a random nonrepeating set of key characters. The key characters are added to the plaintext characters modulo 26; the key is only used once and then discarded [9].

3.2.2 Types of Algorithms

There are two types of cryptographic algorithms: symmetric and public key. In a *symmetric* algorithm, the encryption and decryption keys are known to both sender and receiver [12]. Most of the time the keys are the same and other times they are closely related by a simple transformation. Examples of symmetric algorithms include the DES and the AES. In contrast, a *public key* algorithm uses two distinct keys. One of these keys, called the *public* key, is freely available to any party. The other key, called the *private* key, is kept secret; each party has their own private key that corresponds to the public key. It is virtually impossible for an adversary to deduce the private key in a reasonable amount of time given the public key. In a typical system, the encryption key is the public key and the decryption key is the private key [9]. The most widely known public key cryptosystem is Rivest-Shamir-Adleman (RSA).

Symmetric algorithms can be classified as block ciphers or stream ciphers. In a block cipher, the message is partitioned into predetermined block sizes, fed through the algorithm, output in blocks, and concatenated for the receiver to interpret. In a stream cipher, each character in the plaintext is encrypted separately [13]. Section 3.3 will cover more on the topic of block ciphers, in particular DES.

3.2.3 Keys

The encryption/decryption keys are extremely important to the security of a cipher. Algorithms are generally public knowledge, therefore anyone with a brain can figure out how a plaintext message moves through the algorithm. However, it is a combination of the algorithm complexity and key length that ultimately determine how secure a cryptosystem will be.

If Eve knows the key, then she can read all messages encrypted with that key. Eve could conduct an exhaustive attack by trying all possible keys, but if the key is long enough, this could be infeasible. Therefore, it is generally true that a longer key is more difficult to break than a shorter one. For example, AES uses a variable key length of 128, 192, or 256 bits, where each bit is either a zero or one. Thus, the key space for a 256 bit AES key is 2^{256} possible keys, or roughly 1.1579×10^{77}. For some perspective, the Earth is approximately 4.54 billion years old (4.54×10^9) while the universe is roughly 13.8 billion years old.

From a purely theoretical standpoint, let us assume we have a processor that can perform 10^9 encryptions per second. If a collection of 1000 processors attempts an exhaustive search of all $2^{128} \approx 3.4 \times 10^{38}$ keys for a 128-bit key, then it would take roughly 10^{19} years to search this space. Even if we had access to one of the world's fastest computers in China that operates at 33.86×10^{15} floating point operations per second [14], at 300 operations per encryption this would take roughly over 95 quadrillion years to exhaust the key space.

3.3 Block Ciphers

The history of the term *block cipher* is somewhat vague. Many classical and historical cryptosystems are deemed block ciphers, but the modern-day idea of a block cipher was not cemented until the 1970s. Some examples of early block ciphers include: Vigenère ($\approx$ 1550), Playfair (1854), and Hill (1929). In 1973, the National Bureau of Standards (NBS), the current National Institute of Standards and Technology (NIST), issued a request for a cryptosystem to become the new national standard for encryption. NBS required this standard to be a block cipher, essentially initiating the formal study of block ciphers. DES and AES are the two most common examples of block ciphers.

3.3.1 Definition and Design

Formally, a block cipher is a pair of functions [15] E and D:

$$E : \mathbb{V}_k \times \mathbb{V}_n \to \mathbb{V}_n \tag{3.1}$$

$$D : \mathbb{V}_k \times \mathbb{V}_n \to \mathbb{V}_n. \tag{3.2}$$

In other words, a block of plaintext of bit length n is combined with a key of bit length k, producing an encrypted block of ciphertext of bit length n. Similarly, the decryption function takes an n-bit ciphertext with a k-bit key and maps the combination into an n-bit plaintext. In traditional math lingo, E and D *undo* each other and are thus inverses.

Most modern block ciphers operate in *iterated* fashion, meaning the blocks of plaintext pass through a *round function* f for a set number of rounds. The purpose of this is to increase algorithm security by repeatedly using the same function. Each round uses a different key derived from the previous one, further increasing the security. Figure 3.2 depicts the situation just described.

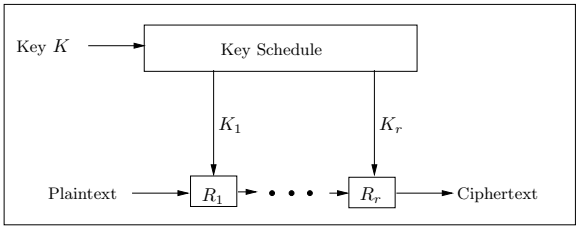


Figure 3.2: General Structure of a Block Cipher, from [15].

There are various ways to design a cryptosystem to achieve an adequate level of security in encryption. The two main design techniques are the *Feistel system* and *substitution-permutation networks (SPN)*. A Feistel system is depicted in Figure 3.3, while SPN is displayed in Figure 3.4.

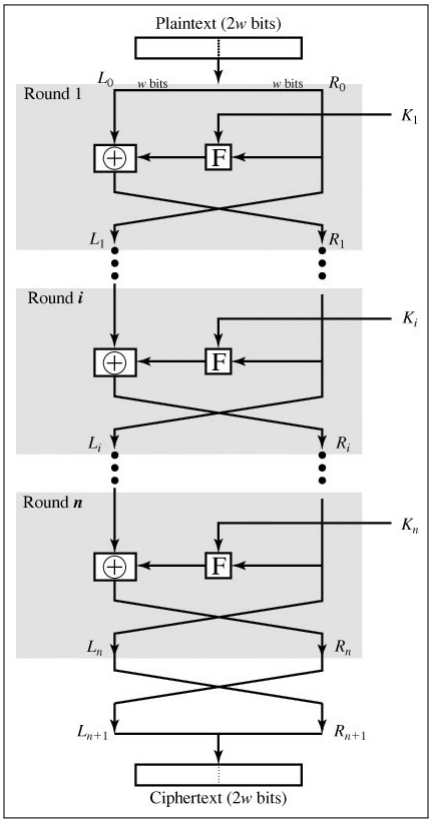


Figure 3.3: General Structure of a Feistel System, from [16].

The Feistel system is named after the German born cryptographer Horst Feistel. In the

Feistel cipher, the first round is initiated with a split of a plaintext block into two halves, called the left and right. The right side and the round key pass through the round function, the result of which is then combined with the left side via the logical exclusive or (XOR) (in binary, this is equivalent to addition *modulo* 2). The result of this XOR then swaps with the preceding right side and becomes the new right side for the next round. This process then iterates over a set number of rounds. After the last round, the resulting left and right parts become the ciphertext block. Since this process must be invertible, decryption works in the same manner but in the the reverse direction.

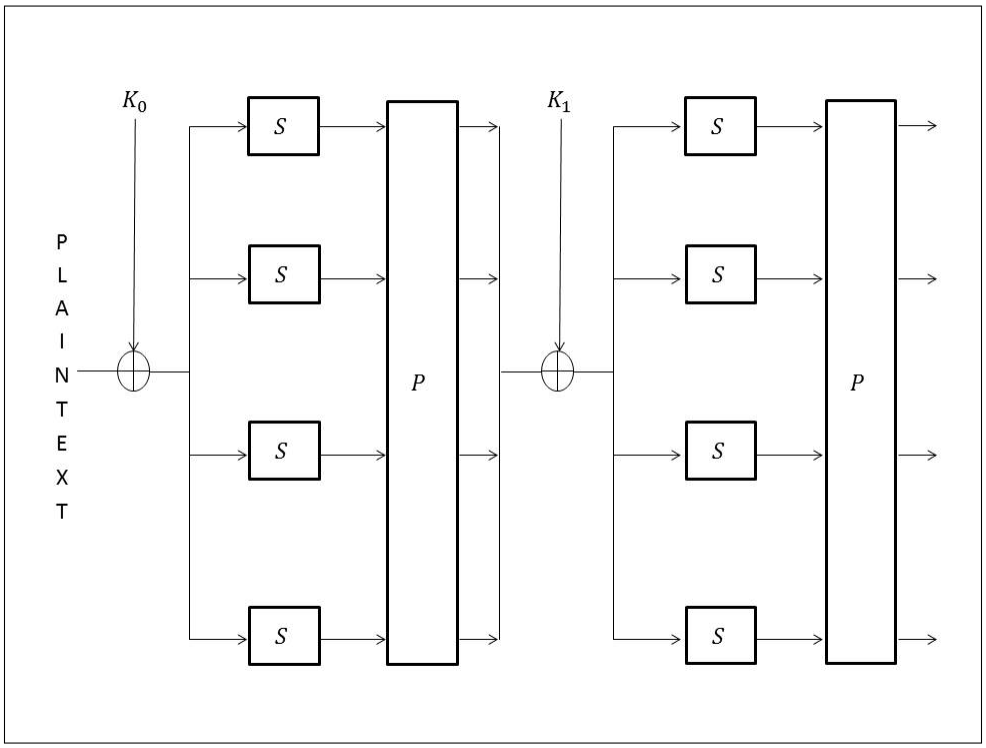


Figure 3.4: Substitution-Permutation Network, after [15].

In the SPN, the encryption algorithm makes use of two basic cryptographic operations: *substitution* and *permutation*. SPNs are a type of *product* cipher because they involve more than one transformation, i.e., substitution and permutation, essentially mixing confusion and diffusion over and over again. The plaintext block and the initial key are combined via XOR, the result of which is then subdivided into smaller blocks and passed through a substitution step. Each of the boxes in Figure 3.4 labeled with an *S* is known as a substi-

tution box (S-Box), and these introduce confusion in the cipher. In the substitution step, each character is replaced with another character. A permutation step follows substitution, in which the bits are permuted or re-ordered. Permutation generates diffusion in the cipher. Following the permutation step, the resulting block is combined with the next round key via XOR and the process iterates.

3.3.2 Advantages and Disadvantages

One of the primary drawbacks to any symmetric algorithm is key distribution [13]. If Alice wants to talk to Bob using a symmetric algorithm, then Alice and Bob need to have the same key. If Alice and Bob are on separate continents, however, key distribution could prove to be difficult. In addition, if Alice wants to talk with Charles, then she needs a different key than the one used to converse with Bob. Key generation is also an issue, but this process will be discussed more in depth in Section 3.4. Block ciphers also present their own advantages and disadvantages as displayed in Table 3.1.

	Block Encryption Algorithms
Advantages	• *High diffusion*. Information from the plaintext is diffused into several ciphertext symbols. One ciphertext block may depend on several plaintext letters. • *Immunity to insertion of symbols*. Because blocks of symbols are enciphered, it is impossible to insert a single symbol into one block. The length of the block would then be incorrect, and the decipherment would quickly reveal the insertion.
Disadvantages	• *Slowness of encryption*. The person or machine using a block cipher must wait until an entire block of plaintext symbols has been received before starting the encryption process. • *Error propagation*. An error will affect the transformation of all other characters in the same block, although there are techniques of self-healing when implementing the block cipher; (See the next section.)

Table 3.1: Analyzing Block Algorithms, after [13].

Additionally, while block ciphers can be used in a variety of modes, they are often more difficult to analyze mathematically than stream ciphers. However, block ciphers are often more suitable for software implementation because they avoid bit by bit computations and work on blocks of information that can be implemented in computers very efficiently [9].

3.3.3 Modes of Operation

Recall that a block cipher operates on a block of plaintext. Issues arise, however, when the message size differs drastically from the block size. For example, a block cipher acting on a block size of 128 bits needs help if the message size is only 20 bits. To account for the varying needs of users and their messages, block ciphers can operate in a variety of *modes*. The most common modes of operation are listed below:

- electronic codebook (ECB)
- cipher block chaining (CBC)
- cipher feedback (CFB)
- output feedback (OFB)
- counter (CTR).

Electronic Codebook Mode

ECB is the most common mode of operation for a block cipher. Given an encryption function E_K, a plaintext block P is subdivided into smaller *words* $P = [P_1, P_2, \ldots, P_L]$ and produces the ciphertext $C = [C_1, C_2, \ldots, C_L]$, where $C_j = E_K(P_j)$ is the encryption of P_j using the key K. In other words, each of the words in the plaintext is encrypted using the same key [10, 12]. Since each plaintext block encrypts independently of another, this mode is easy to work with and favors parallel processing on multiple machines. Additionally, errors in transmission remain within the associated block and do not affect other blocks. However, the major weakness with ECB is that identical blocks of plaintext encrypt to identical blocks of ciphertext. Due to redundancies in most communication, an adversary can detect repetitions and build a codebook without even knowing the key [9].

Cipher Block Chaining Mode

CBC incorporates the method of chaining, a feedback mechanism that resembles a recursive operation. The encryption of a given block depends on the encryption of previous

blocks. Using notation from the previous paragraph, encryption is defined as

$$C_j = E_K(P_j \oplus C_{j-1}). \tag{3.3}$$

Thus, as evidenced in Figure 3.5, the plaintext is XORed with the previous ciphertext block. Equation 3.3 allows for a value of C_0, which is some chosen initial value represented as an *initialization vector (IV)*. The purpose of an IV is to make each message unique, thus alleviating the problem of identical plaintext messages encrypting to the same ciphertext messages [9].

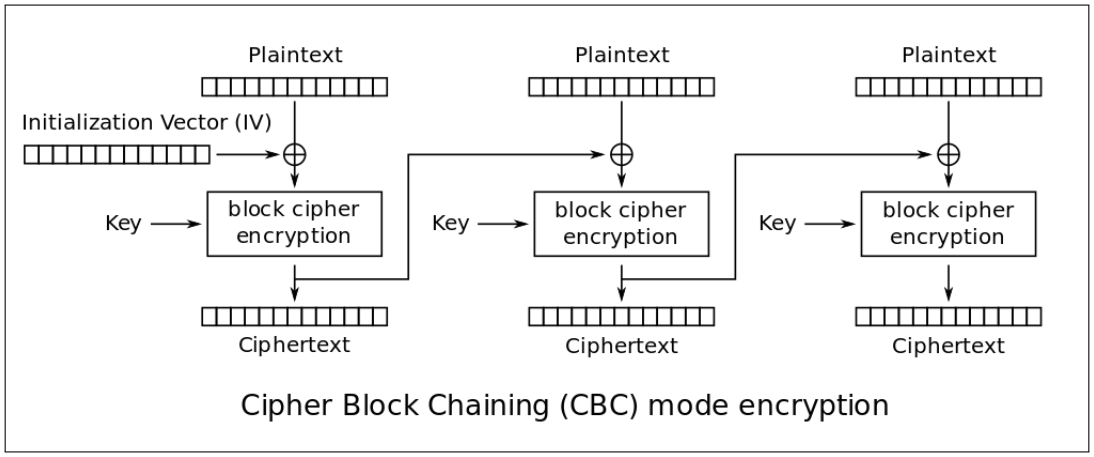


Figure 3.5: Cipher Block Chaining Mode, from [17].

Cipher Feedback Mode

CFB allows for encryption/decryption of a set of characters smaller than the block size. In this sense, CFB is a way to implement a block cipher as a stream cipher. In general, CFB operates on a k-bit mode, where k is less than or equal to the block size. The plaintext $P = [P_1, P_2, \ldots]$ is broken down into k-bit chunks, where each P_j has k bits. Encryption is once again started with an IV, which can be public, but it is unique for each block of encryption. Once the IV is encrypted, the left most k-bits of this result are XORed with the first k-bits of the plaintext. The result of this operation is the first chunk of ciphertext. For the next stream of encryption, this k-bit chunk of ciphertext is then appended to the right side of the IV, shifting all bits k positions to the left (left most k-bits are discarded). Encryption then proceeds in the same manner. Mathematically, encryption is defined for

$j = 1, 2, 3, \ldots$, on an n-bit plaintext message in the following manner:

$$O_j = L_k(E_K(X_j)) \tag{3.4}$$

$$C_j = P_j \oplus O_j \tag{3.5}$$

$$X_{j+1} = R_{n-k}(X_j)||C_j. \tag{3.6}$$

L_k refers to the leftmost k-bits and R_{n-k} refers to the rightmost $n-k$ bits; X_1 is the IV and || refers to concatenation. Figure 3.6 depicts CFB on a s-bit mode.

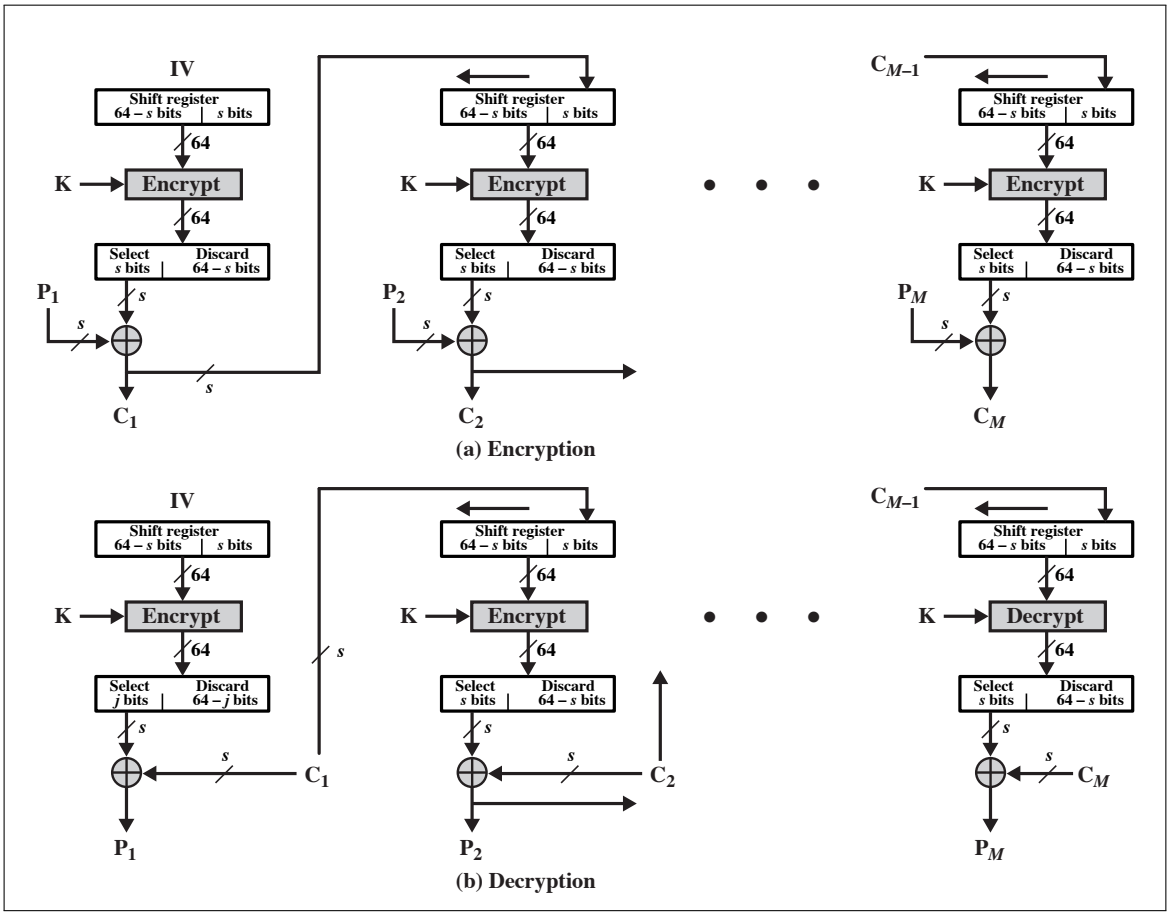


Figure 3.6: s-bit Cipher Feedback Mode on 64-bit Plaintext, from [18].

Output Feedback Mode

Figure 3.7 depicts OFB on a s-bit mode. OFB is another method of implementing a block cipher in a stream mode. Just like in CFB, the IV is encrypted; the leftmost k-bits of this result (call this O_j) are extracted and XORed with the first k-bits of the plaintext, producing the first k-bits of ciphertext. For the next stream, rather than use the ciphertext as the input to the next IV, OFB takes O_j and appends this chunk to the right side. Mathematically, encryption is defined for $j = 1,2,3,\ldots$, on an n-bit plaintext block in the following manner:

$$O_j = L_k(E_K(X_j)) \tag{3.7}$$

$$X_{j+1} = R_{n-k}(X_j)||O_j \tag{3.8}$$

$$C_j = P_j \oplus O_j. \tag{3.9}$$

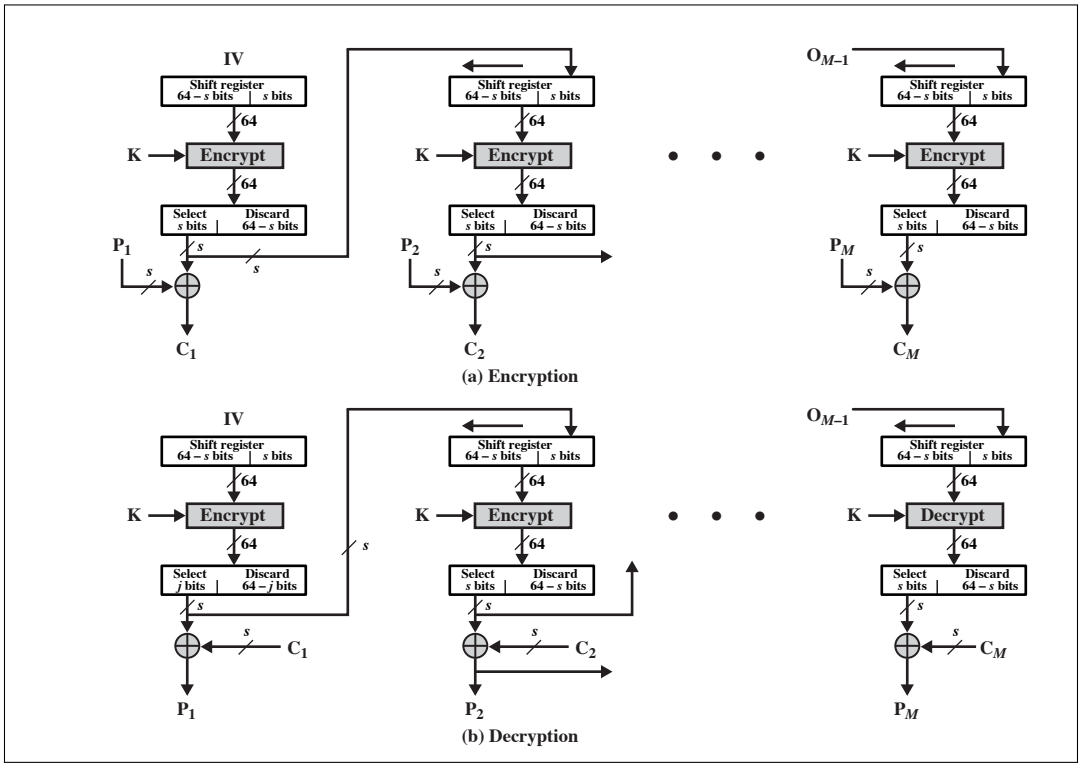


Figure 3.7: s-bit Output Feedback Mode on 64-bit Plaintext, from [18].

The operation in both CFB and OFB involving appending, shifting, and discarding bits is

very similar to the way that a LFSR works. LFSRs can quickly produce a pseudorandom sequence of bits defined by a linear recurrence relation. LFSRs have wide usage, especially in military cryptography and for more on the subject consult [9, 12].

While CFB and OFB operate in similar manners, there are glaring differences with regards to error propagation. In CFB, an error in the plaintext will affect all outputs of ciphertext due to the recurrence relation. An error in the ciphertext, however, can be flushed out since eventually the ciphertext block with the error(s) will be shifted left until discarded. The problem here is that decryption produces nonsensical plaintext until errors are flushed. In OFB, errors in the ciphertext do not propagate; bits of ciphertext that are corrupted translate to corresponding bits in the plaintext with corruption. Since successive rounds are not built using corrupted ciphertext, errors do not repeat into other rounds. OFB can be used offline since future streams do not depend on the plaintext message being present. However, various professionals such as Robert Jueneman have shown that k-bit OFB mode is insecure for values of k less than the block size [19]. The key stream O_j has to eventually repeat, but the concern is that this repeat happens with the same key. When k is equal to the block size n, the cycle length of key streams averages to $2^n - 1$. When $k < n$, this average cycle length drops to $2^{n/2}$, making it a much shorter time to find the repetition [9].

Counter Mode

CTR mode is similar to OFB but the output of the encryption is not used in the next stream. Instead, the encryption input vector is incremented by some constant, typically one, and used in the next register. The mode starts with an IV of length equal to the block length and is encrypted with key K. The leftmost k-bits of this result are XORed with the first k-bit chunk of plaintext to produce the first k-bit piece of ciphertext. A new encryption stream is then created by adding one to the IV and the process iterates. Note how the new vector does not depend on the encryption from the previous output. This process is depicted in Figure 3.8.

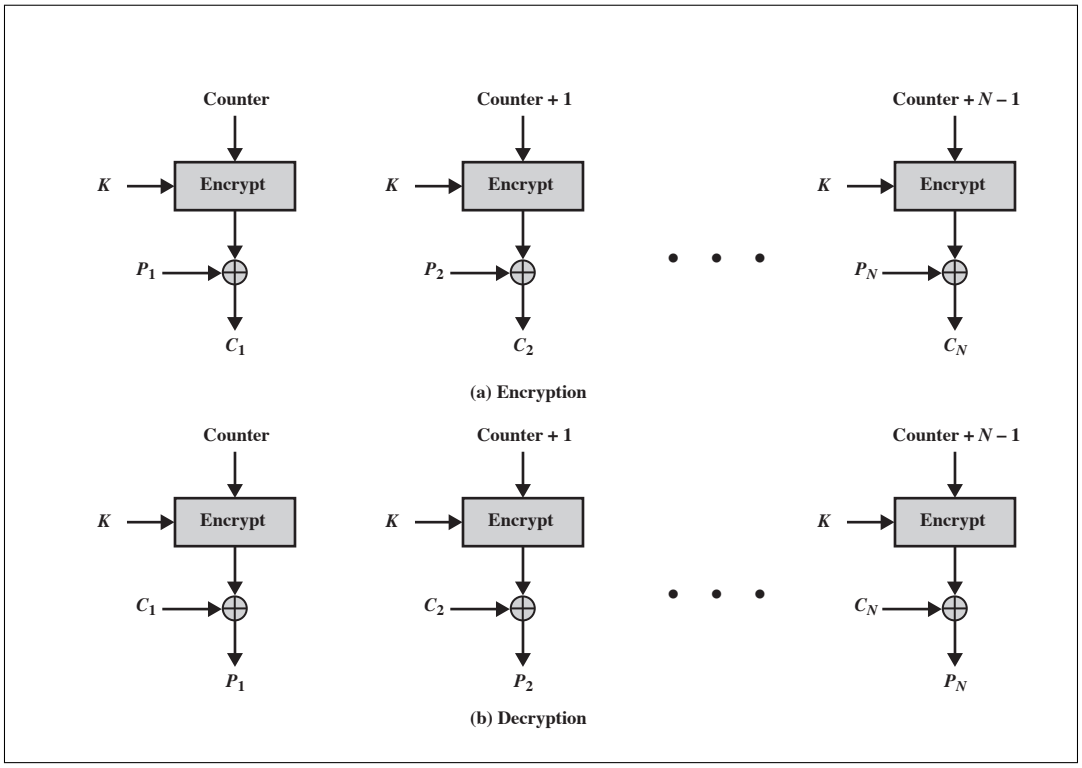


Figure 3.8: Counter Mode, from [18].

Mathematically, encryption in CTR mode is given by

$$X_j = X_{j-1} + 1 \tag{3.10}$$

$$O_j = L_k(E_K(X_j)) \tag{3.11}$$

$$C_j = P_j \oplus O_j. \tag{3.12}$$

3.4 The Data Encryption Standard

DES is perhaps the most well-known block cipher of the last century. It was for several decades the standard for data transmission in electronic commerce. Although it is no longer secure enough for much of our business needs in the United States (U.S.), DES is still in use as a primary system in some parts of the world and even for lower level applications in the U.S. such as secure speech [20].

3.4.1 History

Although cryptographic algorithms have been in use for quite awhile, times of intensive military conflict have necessitated the need for secure communications. The world wars forced militaries to create ciphers to facilitate communication. The breaking of the infamous Zimmermann Telegram accelerated the U.S. entry into WWI. The German Enigma machine was in use for almost 20 years before the British and Polish were able to decrypt its messages in WWII. Claude Shannon gave us further insight into making cryptographic algorithms stronger following the wars, in 1949.[3] Furthermore, with computers coming to the forefront in the 1950s and 1960s, the need to protect data in the commercial sector became apparent [11].

Various private industries began earnest work into the development of strong block ciphers in the late 1960s [11]. Due to wars and the need for protecting government data, cryptology generally fell to the hands of the U.S. Department of Defense and Department of State. The rise in commercial industry, however, engendered the need for a public encryption system to be created. The NBS was charged with the task of finding this algorithm.

At the time, International Business Machines (IBM) was already involved in cryptography and algorithm development. According to D. Coppersmith, IBM was asked in the early 1970s by Lloyd's of London insurance to develop an encryption scheme for protecting automated teller machine (ATM) data [21, 22]. Officially, NBS issued a public request for a national cryptographic standard in the 1973 *Federal Register*. NBS specified nine major design principles, some of which included: ability to provide a high level of security, available to all users, adaptable to multiple applications, exportable, security depending on the key and not the secrecy of the algorithm, etc. [9, 13]. Few products were submitted, and none of them met sufficient criteria for a standard, thus NBS issued a second request in the 1974 *Federal Register*.

IBM was already working on an algorithm when NBS issued their request. At two separate sites (Kingston and Yorktown Heights, NY), the IBM team consisting of Roy Adler, Don Coppersmith, Horst Feistel, Edna Grossman, Alan Konheim, Carl Meyer, Bill Notz, Lynn Smith, Walt Tuchman, and Bryant Tuckerman developed an algorithm they dubbed *Lucifer*

[3] C. Shannon wrote arguably the most influential paper of the 20th century on cryptography in 1949, "Communication Theory of Secrecy Systems."

[9, 13, 21]. IBM submitted *Lucifer* to NBS in 1974, who forwarded the algorithm to the National Security Agency (NSA) for review. After some modifications, NSA returned a version which was approved and published by NBS in 1975 as DES. After two years of critique and criticism, NBS adopted DES as the national standard in 1977 [9, 12].

From its publication in 1975, DES has been embroiled in controversy. First, the proponents of *Lucifer* were dismayed that the NSA reduced the key size from 128 bits to 56. Second, the design considerations of DES were not released at the time of publication. This worried some because many thought that either IBM or the NSA had built a "trapdoor" into the algorithm, i.e., a secret weakness to allow only them to be able to break the system. However, Coppersmith argues that this was not the case; IBM was circumspect and disclosure of this information was to prevent cryptanalysis [21]. Finally, the NSA "characterized DES as one of their biggest mistakes" [9]. The NSA approved the standard with the notion that DES would be a hardware-only protocol; NBS issued the standard with enough information so that programmers could write DES software. In this respect, DES did more for the field of cryptanalysis, and it came to no surprise that the next government standard algorithm (*Skipjack*) was classified [9].

DES was officially published on January 15, 1977, as Federal Information Processing Standards (FIPS) Publication 46. NBS required that the standard be recertified and validated every five years after that. In 1983, DES passed the test easily. In 1988, however, the NSA had objections to the standard and demurred that it would not take long for DES to be broken. Unfortunately, there were no other viable alternatives available and businesses were regularly using DES for encryption needs [9]. The standard was recertified and updated on January 22, 1988, as FIPS Publication 46-2. DES was again recertified in 1993. By 1997, however, several methods were known for attacking DES like systems, thus initiating the search for a replacement. DES was recertified on October 25, 1999, as FIPS Publication 46-3, which also encouraged the use of Triple DES (equivalent of a 112-bit key) to secure data [12]. With successful cryptanalysis occurring in 1999, NBS (now the NIST) convened to select a replacement. Finally, in November 2001 AES was published but DES would remain in place until its removal in May 2005. For almost 30 years, DES was the national standard for encryption.

3.4.2 Algorithm Overview

DES is a symmetric block cipher operating on blocks of 64-bit plaintext. It is a Feistel type system whose round function utilizes SPN operations. The key is 56 bits in length, although it is expressed as a 64-bit string; every eighth bit is a parity check bit used for error detection and is usually ignored (see a text on coding theory for more on this subject). Since encryption must be invertible, a 64-bit block of plaintext encrypts to a 64-bit block of ciphertext. Thus, encryption and decryption can be visualized [23], respectively, as

$$KEY(56\ bits) + Plaintext(64\ bits) = Ciphertext(64\ bits) \tag{3.13}$$

$$KEY(56\ bits) + Ciphertext(64\ bits) = Plaintext(64\ bits). \tag{3.14}$$

Outline

Figure 3.9 depicts the DES algorithm, consisting of 16 rounds. A 64-bit block w of plaintext is sent through an initial permutation (IP), to obtain $w_0 = IP(w)$. This new block is then split into a left and right half, each 32 bits long, i.e., $w_0 = L_0R_0$. For 16 rounds, the operations are the same. The right half goes into the round function f while also becoming the left half of the next round. The left half is XORed with the output of the round function, and the result of this XOR becomes the right half of the next round. Mathematically, this is given for $1 \leq i \leq 16$ as

$$L_i = R_{i-1} \tag{3.15}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i). \tag{3.16}$$

The notation K_i represents the ith key, but only 48 bits from the 56-bit key. After applying the 16th round function, the left and right halves are swapped, then go through an inverse permutation to obtain the ciphertext $c = IP^{-1}(R_{16}L_{16})$.

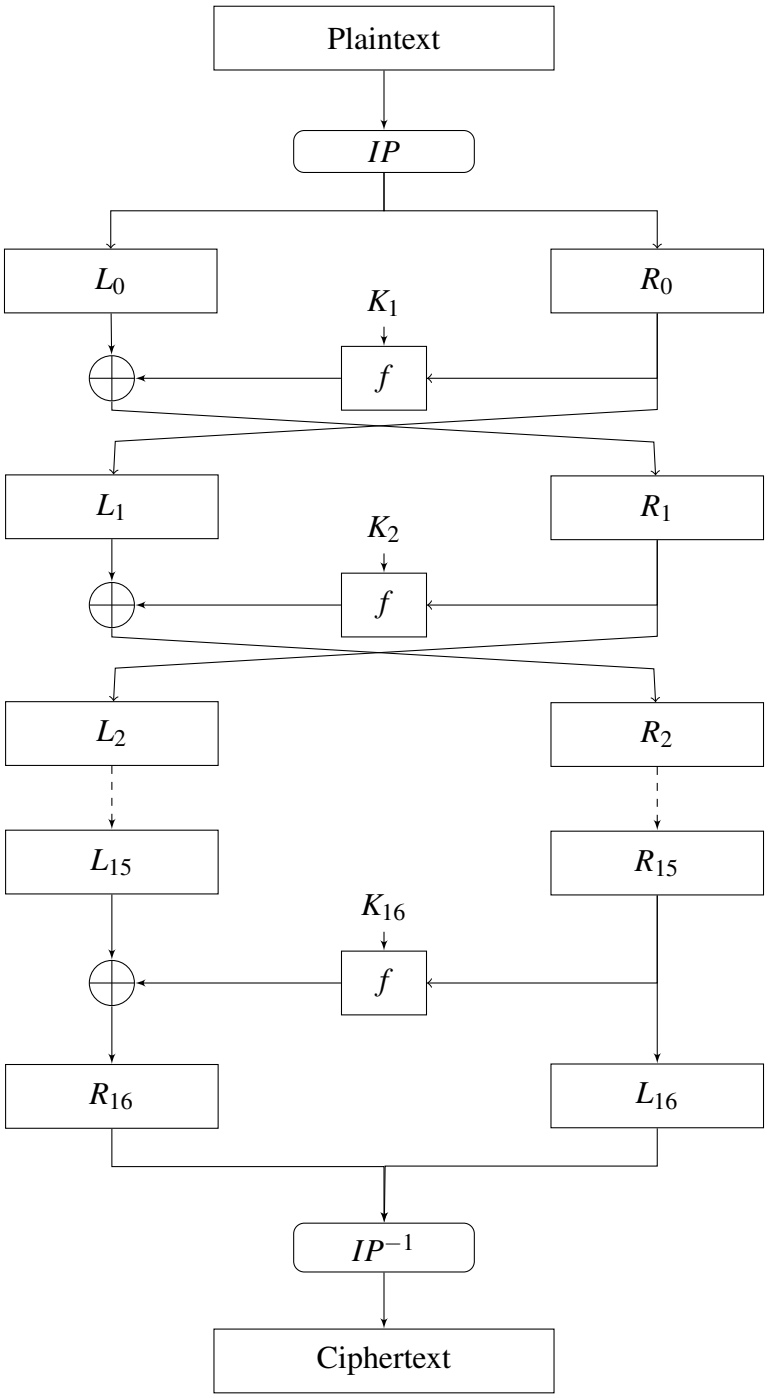


Figure 3.9: The DES Algorithm, after [12].

Initial Permutation

The IP actually occurs before the start of the first round. It does not affect the security of DES, but it also does not have any cryptographic significance. The best explanation is that the IP and inverse IP made data more easily readable by processors in the 1970s [9, 12]. This step is essentially a table look up, read left to right and top to bottom. The IP is listed below in Table 3.2. For example, the 58th bit of w becomes the 1st bit of w_0, the 50th bit of w becomes the 2nd bit of w_0, 42nd bit of w becomes the 3rd bit of w_0, etc.

Initial Permutation															
58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

Table 3.2: DES Initial Permutation, from [12].

Round Function

Recall that the input to each round function is the right half of the block from the previous round. The function f has a number of steps within it, the first of which is another permutation called *expansion*. This expansion permutation is depicted in Table 3.3, whereby R is expanded to $E(R)$. Note that this table has 48 bits of output operating on an input of 32 bits. While the reader will note repetitions in the table, each input block generates a unique output block. The table reads the same as the IP, i.e., the 32nd bit of the input block becomes the 1st bit in the expansion block, etc. The purpose of expansion is not only to provide a block size equal to the key length for the XOR operation, but also to exhibit an *avalanche effect*. In other words, one bit affects two substitutions [9].

Expansion Permutation											
32	1	2	3	4	5	4	5	6	7	8	9
8	9	10	11	12	13	12	13	14	15	16	17
16	17	18	19	20	21	20	21	22	23	24	25
24	25	26	27	28	29	28	29	30	31	32	1

Table 3.3: DES f Expansion Permutation, from [12].

After expansion, $E(R)$ is then XORed with a 48-bit subkey K_i (key generation will be discussed later). The result of $E(R) \oplus K_i$ is another 48-bit string, which is partitioned into 6-bit chunks labeled $B_1B_2 \cdots B_8$. These B_j then go through a substitution step. Substitution is performed via S-Boxes, whereby the input to S_j is B_j. The input to each S-Box is a 6-bit string, while the output is a 4-bit string. Substitution will be discussed in greater detail in the next subsection.

The outputs of the S-Boxes are eight 4-bit chunks, which are concatenated to form $C_1C_2 \cdots C_8$. This new string then goes through another permutation, sometimes known as the *P-Box*. The P-Box permutation is shown in Table 3.4. This operation completes the round function; the layout of the DES round function is displayed in Figure 3.10.

Permutation															
16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

Table 3.4: DES f Permutation, from [12].

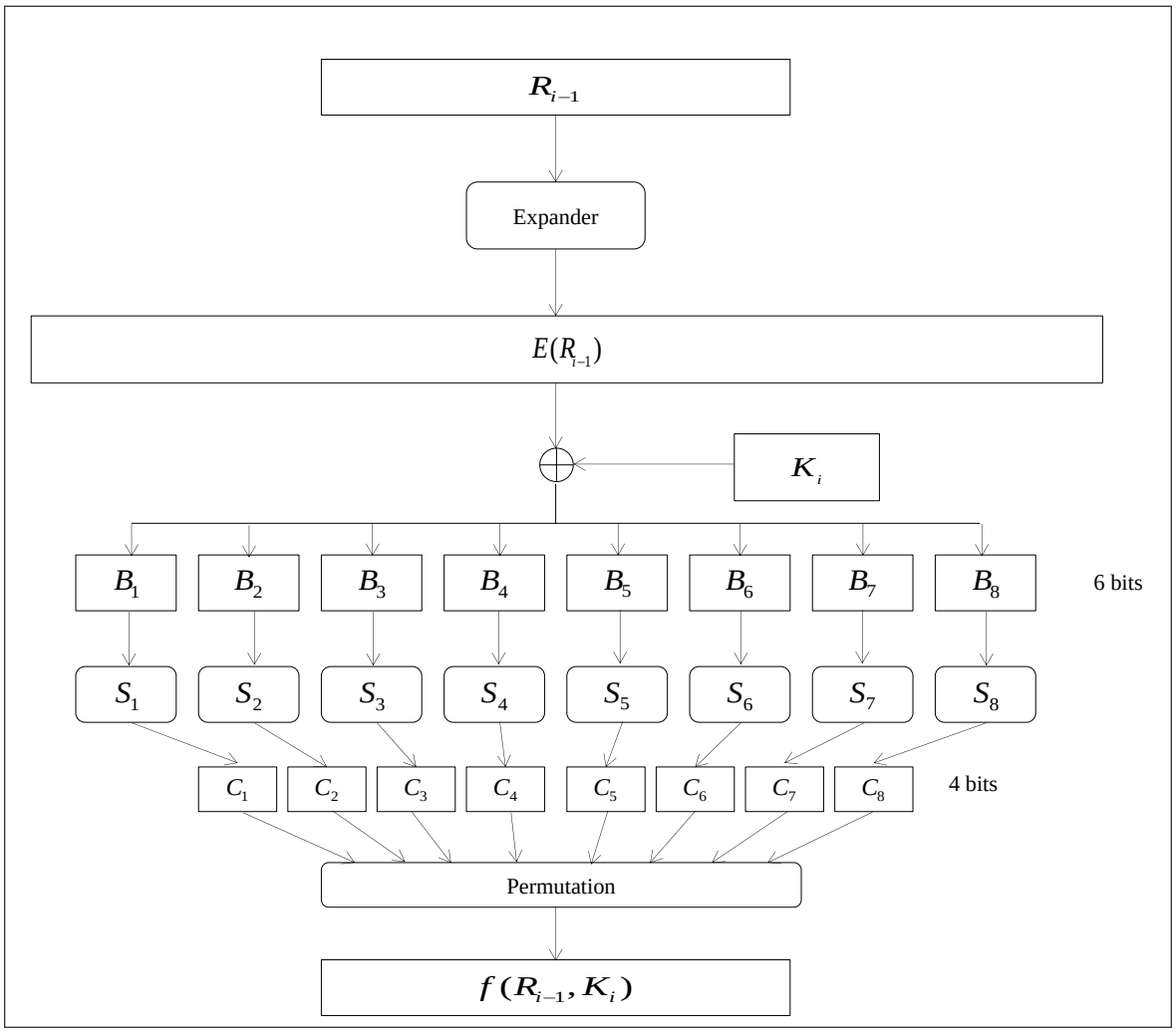


Figure 3.10: The DES Function f, after [12].

Key Generation

Recall that the initial DES key is 64 bits in length, but every eighth bit is a parity check bit. Thus, ignoring the parity check bits, the key is reduced to a 56-bit string K. As was written in the original registers [24–26], the key bits are then permuted via *Permuted Choice-1*. Following the first permutation, the key is split into two halves of 28 bits each, $K = C_0D_0$. C_0 and D_0 then undergo a left shift to obtain C_1 and D_1. Each bit in C_0 and D_0 will shift left one place, but in general this is not the case. In general for $1 \leq i \leq 16$, the left shift is described by $C_i = LS_i(C_{i-1})$ and $D_i = LS_i(D_{i-1})$, where LS_i implies a left shift of one

or two places in the ith round. Both the first permutation and left shift are described in Tables 3.5 and 3.6.

Permuted Choice-1													
57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

Table 3.5: DES First Key Permutation, after [12].

Number of Key Bits Shifted Per Round																
Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Shift	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Table 3.6: DES Key Left Shift Operation, from [12].

After the left shift, the 56-bit string C_iD_i undergoes one final permutation, denoted *Permuted Choice-2*. This second permutation is sometimes also called a *compression permutation* because it selects a subkey of 48 bits from the 56-bit input. The result from Permuted Choice-2 is K_i for each round. This compression is required because the other input to the XOR operation in the round function is the 48-bit expansion string $E(R)$. Permuted Choice-2 is displayed in Table 3.7.

Permuted Choice-2											
14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

Table 3.7: DES Second Key Permutation, after [12].

Inverse Initial Permutation

The final operation in the DES algorithm is another permutation, the inverse of the IP. After the last round, the left and right halves do not swap but instead concatenate to form the input for IP^{-1}. The purpose of IP^{-1} is to ensure that the algorithm can be used for decryption. In decryption, the algorithm performs in the same manner, but the order of the keys is reversed [9, 12]. IP^{-1} is displayed in Table 3.8.

Inverse Initial Permutation															
40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

Table 3.8: DES Inverse Initial Permutation, from [12].

3.4.3 Substitution Boxes

Recall that within the DES round function, the input to the S-Boxes are the blocks $B_1B_2\cdots B_8$. Each of the B_j is assigned to the corresponding S-Box S_j, where S_j is a table lookup. The 6-bit input B_j is written as $b_1b_2b_3b_4b_5b_6$. The end bits b_1 and b_6 are used to determine the row of S_j; $b_2b_3b_4b_5$ determine the column of S_j. The entry in the corresponding row and column of the S-Box is the output. The output of the S-Boxes is $C_1C_2\cdots C_8$, where C_i is a 4-bit string. In this respect, each S-Box acts as a function mapping six bits of input to four bits of output. In fact, the S-Boxes are represented by a special class of cryptographic functions called *Boolean functions* (more on this in Chapter 4).

Table 3.9 displays the first S-Box in its traditional manner. Note that each row in the box contains the numbers zero through 15 exactly once. The reader might wonder how six bits of input will produce four bits of output given this form. Since a bit takes on the value of zero or one, the S-Box needs to be converted to its binary form (see Table 3.10).

S-Box 1																
ROW/COL	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Table 3.9: DES Substitution Box 1, after [12].

S-Box 1								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	1110	0100	1101	0001	0010	1111	1011	1000
01	0000	1111	0111	0100	1110	0010	1101	0001
10	0100	0001	1110	1000	1101	0110	0010	1011
11	1111	1100	1000	0010	0100	0100	0001	0111
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	0011	1010	0110	1100	0101	1001	0000	0111
01	1010	0110	1100	1011	1001	0101	0011	1000
10	1111	1100	1001	0111	0011	1010	0101	0000
11	0101	1011	0011	1110	1010	0000	0110	1101

Table 3.10: DES Substitution Box 1 in Binary Form, after [23].

As a simple example, suppose $B_1 = 001101$. The outer bits $b_1b_6 = 01$ determine the row in the S-Box. The inner bits $b_2b_3b_4b_5 = 0110$ determine the column. Thus, the entry in S_1 is 13, represented as 1101 in binary. This is conveniently colored for the reader in Table 3.10. Concatenating the remaining S-Boxes yields the desired 32-bit string for the next permutation.

The security of the DES algorithm rests primarily in the S-Boxes. For many years, their design was shrouded in mystery and to some extent this is true today. Although the boxes appear to be random shufflings of 32 rows of 16 integers, the IBM design team claims that

the S-Box design is intended to thwart cryptanalysis. To investigate the claims of an alleged NSA trapdoor emplaced in the boxes, the U.S. Senate Select Committee on Intelligence conducted a classified review in 1978 and found no evidence of wrongdoing [9]. Although the findings were not released, the NSA confirmed that they did not tamper with the inner workings of DES. This might appear a closed case on the surface, but several of the IBM designers added further controversy to the topic with their comments. Tuchman and Meyer both stated that the S-Boxes were built by IBM and unaltered by the NSA [9]. Coppersmith stated that the NSA "provided technical advice to IBM" and requested that S-Box design considerations be kept secret [21]. Alan Konheim stated, "We sent the S-boxes off to Washington. They came back and were all different. We ran our tests and they passed" [9]. Clearly, there is some doubt on the veracity of either side of the debate, but an interesting question is why these eight S-Boxes were chosen out of the possible $8!\binom{2^{256}}{8}$.

The NSA has since revealed several design criteria relating to the construction of the DES S-Boxes [27]. They are summarized as follows:

P1. No S-box is a linear or affine function of the input.
P2. Changing 1 input bit to an S-box results in changing at least 2 output bits.
P3. $S(\boldsymbol{x})$ and $S(\boldsymbol{x}+001100)$ must differ in at least 2 bits.
P4. $S(\boldsymbol{x}) \neq S(\boldsymbol{x}+11ef00)$ for any choice of e and f.
P5. The S boxes were chosen to minimize the difference between the number of 1's and 0's in any S-box output when any single output bit is held constant.

Several of the original *Lucifer* designers have also shed some light on the selection and design of the S-Boxes. Meyer wrote that as the number of design criteria increased, the selection of the appropriate S-Boxes was based on the number of terms in the corresponding boolean expressions [11]. According to Meyer, in order to enable implementation on a single logic chip, it was necessary to keep the number of terms around 52 and 53. Coppersmith also wrote a detailed explanation of the eight S-Box design principles that were used in the original specifications. These criteria are listed below:

S-1 Each S-box has six input bits and four output bits (largest size at the time to put on a chip).
S-2 No output bit should be too close to a linear function of the input bits (output bits

cannot be a linear combination of the input bits over $\mathbb{F}_2$).

S-3 Each possible 4-bit output is attained exactly once as the middle four input bits range over their 16 possibilities.

S-4 If two inputs differ in exactly one bit, then the outputs must differ in at least two bits.

S-5 If two inputs differ in the two middle bits exactly, then the outputs must differ in at least two bits (if $\Delta I_{i,j} = \mathbf{001100}$, then $|\Delta O_{i,j}| \geq 2$).

S-6 If two inputs differ in their first two bits and are identical in their last two bits, then the two outputs must not be the same.

S-7 For any nonzero 6-bit difference between inputs, $\Delta I_{i,j}$, no more than eight of the 32 pairs of inputs exhibiting $\Delta I_{i,j}$ may result in the same output difference $\Delta O_{i,j}$.

S-8 The case $\Delta O_{i,j} = 0$ follows (S-7) but with stronger restrictions.

There are many similarities between the NSA list and Coppersmith's, the most important property being nonlinearity. Linearity will be discussed more in Chapter 4, but a linear algorithm is trivially broken. If an adversary knows a few pairs of plaintext and ciphertext in a linear algorithm over the same field, the key can be recovered by solving a simple linear system.

It is true that generic S-Boxes are chosen to resist differential and linear cryptanalysis. They are usually the only nonlinear part of a cipher, which harkens back to the DES design criteria. Although the S-Box itself is a lookup table, for DES it is a function mapping six input bits to four output bits. In this sense, "larger" S-Boxes are generally more resistant to statistical cryptanalysis [9]. "Larger" in this sense means a greater number of input and output bits associated with the mapping. The selection of S-Boxes in a cipher is a debatable issue. The DES designers claimed that months of analysis went into the selection of the eight S-Boxes. Yet, a randomly designed S-Box can often achieve an adequate level of resistance to attacks. While intentionally designed S-Boxes typically show strong resistance to known attacks, their performance against unknown attacks is unknown. On the other hand, randomly selected S-Boxes of large size can provide an adequate level of security [9].

3.4.4 Cryptanalysis of DES

As was mentioned in Subsection 3.4.1, the security of DES has always been in question. The key space was obviously an immediate issue. With a 128-bit key, the key space is $2^{128} \approx 3.4 \times 10^{38}$, but with a 56-bit key the key space is much smaller at $2^{56} \approx 7.2 \times 10^{16}$. Although this is still a large number, famous cryptographers Whitfield Diffie and Martin Hellman (best known for their invention of public-key crypto) analyzed the results of a *brute force* attack in 1976 [9, 28, 29]. In a brute force attack, the cryptanalyst tries every possible key until ciphertext decrypts to meaningful plaintext. Diffie and Hellman theorized that a special parallel computer costing roughly $20 million could search the entire DES key space in 10^5 seconds, or about one day [28, 29]. Even though Diffie and Hellman acknowledged that this type of attack was only feasible for organizations like the NSA, they predicted that DES would be totally insecure by 1990 [9].

Hellman independently proposed another attack known as a *chosen plaintext* attack in 1980. In a chosen plaintext attack, the adversary is assumed to have control of the cipher but not the key. Thus, he can encrypt any number of plaintext messages and try to use the corresponding ciphertexts to find the key. In Hellman's method, the cryptanalyst needs memory space to store the possible encryptions, and he can thus reduce the time to find the key. A single plaintext block is encrypted under all possible keys, with all 2^{56} results being stored in memory. Then the cryptanalyst only has to insert the plaintext into the cipher, recover the corresponding ciphertext and look the key up in memory. Hellman proposed that a special computer could do this for $4-5 million, yielding 100 solutions per day [9,28].

Israeli cryptographers Eli Biham and Adi Shamir were the first to publicly announce the method of *differential cryptanalysis* in 1990. At the time, brute force was the best known possible attack against DES. Coppersmith argues that IBM knew of this technique and purposely designed the algorithm to defeat this technique. Regardless, differential cryptanalysis is another version of a chosen plaintext attack and it revolutionized the field of cryptanalysis. In this method, the cryptanalyst starts with two plaintext messages p and p'. These messages have a known *difference*, whereby the difference between two strings is found by the XOR, i.e., $\Delta p = p \oplus p'$. Then the cryptanalyst can find the corresponding ciphertext blocks c and c', that also have a known difference Δc. Knowing this difference in ciphertext pairs allows the cryptanalyst to assign probabilities to different keys since

more pairs give information about the most probable key. Specifically, since we know the plaintext and ciphertext differences, then we also know the difference in the strings after the key mixing XOR step (since the XOR cancels the key out when looking at the differences). Knowing this difference, call it ΔA, we can infer differences in the strings following the S-Boxes based on probabilities. These two differences give information about the key [9, 21]. As a toy example for why this works, consider Example 3.4.1.

EXAMPLE 3.4.1. Assume that for some block cipher, the cryptanalyst Eve has access to two messages p and p'. She runs these through the expansion box and arrives at $p = 01101$ and $p' = 11100$. Thus, she can easily calculate the difference between these, i.e., $01101 \oplus 11100 = 10001$. She then runs these blocks through the key mixing step (reminder Eve does not know the key), yielding: $01101 \oplus K_i = 10010$ and $11100 \oplus K_i = 00011$. Eve then calculates the difference between these two outputs: $10010 \oplus 00011 = 10001$. Thus, Eve does not need any information about the key to obtain this. Now she can run the blocks through the S-Boxes and obtain this difference, as well as through the P-box and get this difference. Knowing all these differences allows Eve to run more messages through the cipher and observe which of these are more probable than others, and she can start guessing at keys.

Biham and Shamir first utilized differential cryptanalysis on some reduced-round DES variants. For a six-round DES, they showed that a chosen plaintext attack broke the algorithm in less than 0.3 seconds on a personal computer (pc) [28, 30]. If the encryption machine is not known, but the plaintext-ciphertext pair is known (called a known plaintext attack), then differential cryptanalysis reduces the space to 2^{36} ciphertexts. Biham and Shamir also proved that "any reduced variant of DES is breakable by a chosen plaintext attack faster than via exhaustive search" [28]. A brute force attack on DES requires 2^{55} operations, but Biham and Shamir broke DES with differential cryptanalysis using a chosen plaintext attack on 2^{47} plaintexts. Only 2^{36} ciphertexts are needed, however, to analyze and deduce the key. A known-plaintext attack on DES does not reduce the operation space. While a differential cryptanalytic method might seem like a massive breakthrough in cracking DES, this space is still unreachable in a feasible time period for most people and the costs are high. In fact, if an exhaustive key search of 2^{55} operations is performed, assuming the DES algorithm can be implemented at a modern rate of 1.6 gigabytes/sec, then a chip can

perform $(1.6 \times 10^9)/64 = 2.5 \times 10^7$ DES computations per second. Even at this rate, this would take $2^{55}/(2.5 \times 10^7) \approx 1.4 \times 10^9 \approx 45$ years [31]. Even in a chosen plaintext attack with the ability to store the entire search space, the storage of 2^{44} plaintext-ciphertext pairs for example requires upwards of 280 terabytes (TB) [31]. For some perspective, the highest capacity hard drive on the commercial market right now has a 12 TB capacity and it costs over $1,600.

At the CRYPTO '93 Rump Session, researcher Michael Wiener proposed a design for a theoretical DES brute force cracker that could break the algorithm in an average of 3.5 hours with guaranteed results in seven hours [9]. Wiener estimated the cost of this machine to be $1 million; the machine could conduct a key search in parallel so that 16 encryptions could occur simultaneously [10]. Although no one has publicly admitted to constructing such a machine, this financial cost would not be that expensive for a large organization, government, military, or country.

In 1994, Mitsuru Matsui developed a new cryptanalytic technique called *linear cryptanalysis*. In his first paper, where he developed the method, Matsui reduced the search space to 2^{47} known plaintexts [32]. While this equaled the work of Biham and Shamir, Matsui improved the technique in his second paper and showed a complexity 2^{43} [33]. This method was apparently unknown to the DES designers.

Linear cryptanalysis is a known plaintext attack that essentially makes use of a linear function of the input bits. There are two parts to linear cryptanalysis, which Matsui refers to as *Algorithm 1* and *Algorithm 2* [32]. The goal is to find a linear expression

$$p_1 p_2 p_3 \cdots p_m \oplus c_1 c_2 c_3 \cdots c_m = k_1 k_2 k_3 \cdots k_m, \tag{3.17}$$

where the p_i, c_i and k_i are bit positions in the corresponding plaintext, ciphertext, and key, respectively, such that the expression holds with probability $p \neq 0.5$. The first step entails finding linear equations or approximations relating bits of the plaintext, ciphertext, and key via the S-Boxes. Once this linear relation is determined, the relation is then expanded to the other operations in the cipher to arrive at a linear approximation for the entire cipher. For example, perhaps the second bit of the plaintext XORed with the first and third bits of the ciphertext equal the fifth bit of the key, i.e., $p_2 \oplus c_1 \oplus c_3 = k_5$. However, since the key

is unknown, the algorithm is initiated by setting the right hand side of Equation 3.17 equal to 0 or 1. Thus, we often start with the linear equation $p_1 \oplus p_2 \oplus \cdots \oplus c_1 \oplus c_2 \oplus \cdots = 0$.

Once the expression is determined, the cryptanalyst applies all possible input and output values to the expression to determine the probability the equation is true. By counting the number of times that this equation is true for a given key bit value, we can deduce partial key bits based on probability. Specifically, we find T_{max} and T_{min}, where these represent the maximum and minimum number of plaintexts such that the left hand side of Equation 2.17 is zero. If $|T_{max} - \frac{N}{2}| > |T_{min} - \frac{N}{2}|$, then the partial key guessed is 0; if the inequality is flipped, guess 1 [32, 33]. This guess acts on the notion that for a given key bit value, this T value is the most likely set of bits and the corresponding linear approximation holds with high probability. Although linear cryptanalysis reduces the complexity to 2^{43}, it is still highly theoretical and costly in time, money, and processing power.

A more recent development with linear cryptanalysis was conducted by Pascal Junod in his master's thesis. By implementing Matsui's algorithm on a special processor optimized for linear cryptanalysis, Junod showed via experiment that given 2^{43} known plaintext-ciphertext pairs, the complexity of attack could be reduced to 2^{40} [34].

Still, it would seem that the most popular approach to the cryptanalysis of DES is an exhaustive search of the key space. In 1997, RSA Data Security issued a public challenge to decrypt a DES message and find the key, while also offering $10,000 to the winner. Computer scientist Rocke Verser took on the challenge and submitted the correct key in five months. Verser's method included creating a program to search the key space that thousands of personally and corporate owned computers enlisted processing time on [12].

In 1998, the second challenge was issued by RSA Data Security, but this time the key was found in just 39 days. Later that year, the Electronic Frontier Foundation (EFF) started a project called "DES Cracker" in the summer of 1998, a computer built specifically for parallel computing. For just $250,000, EFF used DES Cracker to find a key in 56 hours [10]. In 1999, RSA Labs issued the third challenge which was won by the DES Cracker again. With 100,000 computers networked across the globe, the correct key was found in 22 hours and 15 minutes, testing over 245 billion keys per second [10]. This essentially spelled the end of DES as a national standard. For more information on how EFF designed

and implemented the DES Cracker, the reader should consult [12].

While brute force attacks as well as linear and differential cryptanalysis tackle the algorithm head on, there are other means to attack DES with known weaknesses. One such way depends on the key used. Some keys are better than others, and specifically a key made up of all 0s or all 1s or a 50/50 split is considered weak. Due to the method for key generation, a key with this makeup will be the same key used in every round of the algorithm [9].

The other potential weakness is in the actual design of the S-Boxes. Several analysts have studied the S-Boxes and shown interesting relationships. Davio et al. expanded on a point that Hellman made concering the redundancy in the fourth S-Box, S_4. S_4 uses only one nonlinear function, and as a result, the last three output bits "can be derived from the first one by complementing some of the input bits and by complementing the second and third outputs under control of the variable x_6" [35]. Desmedt et al. proved that if the input to three neighboring S-Boxes was changed, then the output of the round function f will remain the same under certain conditions. In this set of conditions, the notation $abcdef$ represents the 6-bit input to the S-Boxes [36]. The conditions listed below must all be satisfied:

1. complement the inputs a, b and e of the middle three S-Boxes;
2. complement the input c or d of the last S-Box;
3. do not complement the input f of the middle three S-Boxes.

Additionally, Shamir noted that by examining the XOR of the output bits, there was a clear imbalance. Take for example, S_1, denoted in Table 3.10. If we look at the entries where $s_1 \oplus s_2 \oplus s_3 \oplus s_4 = 0$, where s_i is a bit in the S-Box output, then there are seven such outputs on the left half of S_1 versus 25 on the right half [9, 37]. Similar such imbalance is apparent in the remaining S-Boxes. These are just features of the S-Boxes that an adversary could potentially take advantage of.

CHAPTER 4:
Boolean Functions

I am now about to set seriously to work upon preparing for the press an account of my theory of Logic and Probabilities which in its present state I look upon as the most valuable if not the only valuable contribution that I have made or am likely to make to Science and the thing by which I would desire if at all to be remembered hereafter...

$\sim$ George Boole in a letter to William Thomson, 1851

The study of BFs is a relatively old discipline dating back to the 1800s. The study of BFs in cryptography, however, is fairly nascent. BFs owe their name to English mathematician George Boole (1815-1864). Boole came from a poor, working class family that often struggled to make ends meet. The young Boole became interested in learning and even taught himself Greek by the age of 14. Boole was forced into work at the age of 16, and subsequently became a teacher at a small school in 1831. From that point forward, he remained in academia until his death in 1864. Boole's most significant contribution to mathematics centered on two publications in 1847 and 1854, in which he introduced algebra into Aristotelian logic. The resulting *Boolean algebra* became a building block of modern day circuit analysis and model theory. The definitive work on Boole's life is Desmond MacHale's *George Boole: His Life and Work*, 1985, but a more concise synopsis is available in [38].

4.1 Boolean Algebra and Operations

Perhaps the reader is familiar with the Boolean algebra used in logic and circuit design. This algebra has two operations, namely addition and multiplication on the set $\{0, 1\}$. The Boolean sum and product are given by Table 4.1.

x_1	x_2	$\vee$ (OR)
0	0	0
0	1	1
1	0	1
1	1	1

x_1	x_2	$\cdot$ (AND)
0	0	0
0	1	0
1	0	0
1	1	1

Table 4.1: Boolean Sum and Product Tables.

These operations should not be confused with the ones we define for BFs. BFs also utilize a sum and a product, but they operate on vectors and not just single bits. While the product operation is the same, addition of BFs uses the XOR and has the truth table representation in Table 4.2 (note this is the same as addition in the finite field $\mathbb{F}_2$).

x_1	x_2	$\oplus$ (XOR)
0	0	0
0	1	1
1	0	1
1	1	0

Table 4.2: Boolean Function Addition.

For the world of BFs, we consider a vector space $\mathbb{V}_n$ of dimension n over the two-element field $\mathbb{F}_2$. Thus, elements of $\mathbb{V}_n$ are vectors with n components or in our case bits. We also require this vector space to operate over $\mathbb{F}_2$. Given two vectors in $\mathbb{V}_n$, say $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$, we define addition over $\mathbb{F}_2$ as [39]:

$$\boldsymbol{a} \oplus \boldsymbol{b} = (a_1 \oplus b_1, a_2 \oplus b_2, \ldots, a_n \oplus b_n). \tag{4.1}$$

The bold font is only used to emphasize that these are vectors, but the notation $\vec{a}$ or $\bar{a}$ is sometimes also used. Likewise, we also define the scalar product of two vectors in $\mathbb{V}_n$ as:

$$\boldsymbol{a} \cdot \boldsymbol{b} = a_1 b_1 \oplus a_2 b_2 \oplus \cdots \oplus a_n b_n. \tag{4.2}$$

There is one more operation on BFs that we consider. This operation, denoted by $\star$, resembles a concatenation. This is defined as $\boldsymbol{a} \star \boldsymbol{b} = (a_1 b_1, a_2 b_2, \ldots, a_n b_n)$. We can now define just exactly what a BF is.

4.2 Definitions and Representations

Definition 4.2.1. [39] A Boolean function f in n variables is a map from $\mathbb{V}_n$ to $\mathbb{F}_2$,

$$f : \mathbb{V}_n \to \mathbb{F}_2. \tag{4.3}$$

Since the vector space $\mathbb{V}_n$ is over the finite field $\mathbb{F}_2$, the vectors in the domain of a BF are binary vectors. Thus, $\mathbb{V}_n$ can also be represented as the set $\mathbb{F}_2^n$ of all binary vectors of length n considered as an $\mathbb{F}_2$ vector space [40]. Given this alternate notation, other representations of a BF are

$$f : \mathbb{F}_2^n \to \mathbb{F}_2 \quad [40] \tag{4.4}$$

$$f : \mathbb{Z}_2^k \to \mathbb{Z}_2 \quad [41]. \tag{4.5}$$

It is often more convenient to use the notation given in Equation 4.4, thus we will stick with this for the remainder of the thesis. A BF can be uniquely represented by its *truth table*, a (0,1)-sequence defined as $(f(\boldsymbol{v}_0), f(\boldsymbol{v}_1), \ldots, f(\boldsymbol{v}_{2^n-1}))$, where the $f(\boldsymbol{v}_i)$ are the function output values and the $\boldsymbol{v}_i$ are ordered *lexicographically* [39].

EXAMPLE 4.2.2. Consider the truth table for the BF, $f : \mathbb{F}_2^3 \to \mathbb{F}_2$ in Table 4.3. The unique representation for this BF is given by the column of outputs as a sequence, $(0,0,1,1,1,1,0,1)$. Note that this output column is a binary string of length 2^3.

x_3	x_2	x_1	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Table 4.3: Truth Table of a BF.

Example 4.2.2 displays the truth table representation for a BF, but it deserves some more explanation. A vector in $\mathbb{F}_2^n$ has n bits, and we label the input bits as x_i for $1 \leq i \leq n$. The ordering of the x_i is unimportant; we can order them left to right or right to left. Each row in the truth table represents a vector in $\mathbb{F}_2^n$, and ordering here is important. The vector space $\mathbb{F}_2^n$ contains 2^n vectors, whereby each vector $\boldsymbol{v}_i$ is displayed in a truth table by its binary representation $\boldsymbol{b}(i)$ of i, $0 \leq i \leq 2^n - 1$. Thus, in Table 4.3, the eight vectors in $\mathbb{F}_2^3$ are ordered lexicographically by their binary representations from zero to seven.

The other way to represent a BF is via a polynomial in

$$\mathbb{F}_2[x_1, x_2, \ldots, x_n]/(x_1^2 - x_1, x_2^2 - x_2, \ldots, x_n^2 - x_n).$$

This polynomial representation of a BF is referred to as the *algebraic normal form (ANF)*, given as

$$f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_n) = \sum_{\boldsymbol{a} \in \mathbb{F}_2^n} \lambda_{\boldsymbol{a}} \left(\prod_{i=1}^{n} x_i^{a_i} \right), \quad \lambda_{\boldsymbol{a}} \in \mathbb{F}_2, \quad \boldsymbol{a} = (a_1, a_2, \ldots, a_n). \tag{4.6}$$

Equation 4.6 [42] will make more sense in a bit, but first we need to define some more terms. The *Hamming weight* of an arbitrary vector in $\mathbb{F}_2^n$, denoted by $wt(\boldsymbol{x})$, is the number of 1s in the vector $\boldsymbol{x}$. Similarly, the Hamming weight of f is the number of 1s in the truth table

output sequence. The *support* (or *on-set*) of a BF f, denoted by $\Omega_f = \{\boldsymbol{x} \in \mathbb{F}_2^n : f(\boldsymbol{x}) = 1\}$, is the set of vectors whose truth table output is 1 [39, 42]. Thus, we can also define the Hamming weight of f as $wt(f) = |\Omega_f|$. The *Hamming distance* between two functions f and g is the weight of $f \oplus g$, i.e., $wt(f \oplus g)$.

The *algebraic degree* of f is the largest value of the Hamming weight of $\boldsymbol{a}$ such that $\lambda_{\boldsymbol{a}} \neq 0$ [42], or more simply the number of variables in the highest order monomial with nonzero coefficient [39].

EXAMPLE 4.2.3. Let us refer back to Example 4.2.2 for demonstration of these concepts. Below are the truth table and ANF for this function f. The Hamming weight of f is $wt(f) = 5$; the degree of f is $deg(f) = 3$ since the largest term in the ANF is $x_1x_2x_3$.

x_3	x_2	x_1	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

ANF is $x_2 \oplus x_3 \oplus x_1x_2x_3$

Table 4.4: Representations of a BF.

There is an injective mapping from the ANF representation of a BF to its truth table, so that given one we can find the other. There are several ways to do this, and we start with the algebraic method. The ANF of a BF is specified by its support in the following manner:

$$f(x_1, x_2, \ldots, x_n) = \sum_{\boldsymbol{\tau} \in \Omega_f} \left(\prod_{i=1}^{n} x_i + \tau_i + 1 \right), \quad \boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_n). \tag{4.7}$$

Using Equation 4.7, we can see how the ANF of f was computed in Example 4.2.3. Only the vectors in the support are considered for the ANF. In the expansion below, there is no

difference between the usual "+" and $\oplus$; they both represent the XOR operation, but merely help to differentiate between vectors.

$$\begin{aligned}
ANF &= (x_1+1)x_2(x_3+1) \oplus x_1x_2(x_3+1) \oplus (x_1+1)(x_2+1)x_3 \oplus x_1(x_2+1)x_3 \oplus x_1x_2x_3 \\
&= (x_1+1)(x_2x_3+x_2) \oplus x_1x_2 + x_1x_2x_3 \oplus (x_1+1)(x_2x_3+x_3) \oplus x_1x_3 + x_1x_2x_3 \oplus x_1x_2x_3 \\
&= x_1x_2x_3 + x_2x_3 + x_1x_2 + x_2 \oplus x_1x_2 + x_1x_2x_3 \oplus x_1x_2x_3 + x_2x_3 + x_1x_3 + x_3 \oplus x_1x_3 + x_1x_2x_3 \oplus x_1x_2x_3 \\
&= \cancel{x_1x_2x_3} + \cancel{x_2x_3} + \cancel{x_1x_2} + x_2 \oplus \cancel{x_1x_2} + \cancel{x_1x_2x_3} \oplus \cancel{x_1x_2x_3} + \cancel{x_2x_3} + \cancel{x_1x_3} + x_3 \oplus \cancel{x_1x_3} + \cancel{x_1x_2x_3} \oplus x_1x_2x_3 \\
&= x_2 \oplus x_3 \oplus x_1x_2x_3
\end{aligned}$$

To convert back to the truth table sequence from the ANF, the process is the same with a minor difference. Form a table similar to a truth table but replace the output column with the ANF coefficients. Note in Table 4.5 that in the c column, 1s appear in the rows representing the terms in the ANF $\rightarrow x_2, x_3$, and $x_1x_2x_3$. Reproducing the method from the preceding paragraph will yield the truth table output sequence for the function f.

x_3	x_2	x_1	c
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Table 4.5: Conversion from ANF to Truth Table Sequence.

The other, somewhat quicker method to convert between the two representations is the *Transeunt triangle* as proven by Shafer et al. in [43, 44]. In this method, either the truth table output sequence or ANF sequence is placed in a row. Then in an inverted Pascal's triangle fashion, the consecutive values in this row are added **mod** 2 (synonymous with $\oplus$). The result of the addition is placed in the next higher row between the two values in which the operation was performed [43]. The operations are exhausted until a row with one

entry is reached; the left side of the resulting triangle is the (0,1)-sequence of the desired conversion representation. In Figure 4.1, the function f output from Example 4.2.2 is placed on the bottom row of the Transeunt triangle. After the triangle is formed, the left side is the (0,1)-sequence of ANF coefficients, which matches the polynomial in Example 4.2.3. In an analogous way, if the ANF coefficients are placed on the bottom row, the resulting triangle will reveal the truth table output sequence.

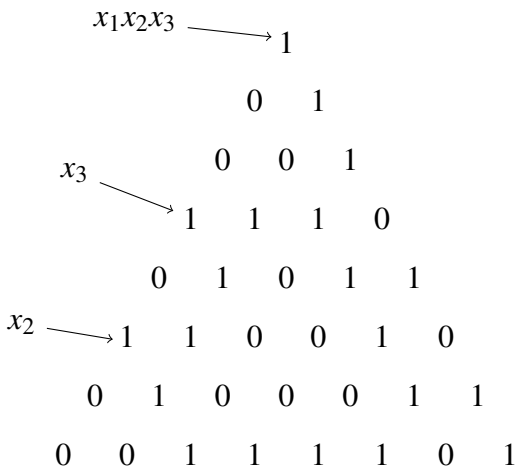


Figure 4.1: Transeunt Triangle Representation.

A BF whose algebraic degree does not exceed one is called an *affine* function. An affine function with constant term equal to zero is called a *linear* function [42, 45]. Mathematically, an affine function on $\mathbb{F}_2^n$ has the form

$$\ell_{\boldsymbol{a},c}(\boldsymbol{x}) = \boldsymbol{a}\cdot\boldsymbol{x} \oplus c = a_1x_1 \oplus \cdots \oplus a_nx_n \oplus c, \tag{4.8}$$

where $\boldsymbol{a} = (a_1, a_2, \ldots, a_n) \in \mathbb{F}_2^n$, $c \in \mathbb{F}_2$ [39].

EXAMPLE 4.2.4. An example of an affine function, $f : \mathbb{F}_2^4 \to \mathbb{F}_2$, is $x_1 \oplus x_2 \oplus x_4 \oplus 1$, while an example of a linear function is $x_1 \oplus x_2 \oplus x_4$.

A BF is called *homogeneous* if its ANF contains terms all of the same degree. The linear function in Example 4.2.4 is homogeneous. A function such as $x_2x_4x_5 \oplus x_1x_3x_5 \oplus x_3x_4x_5$ is also homogeneous.

4.3 Cryptographic Properties of Boolean Functions

Until this point, we have discussed the meaning of a BF and even hinted at nonlinear components of a cryptosystem. Now we need to formally define properties of BFs that make them useful for cryptography. BFs are used in many symmetric key algorithms, and there is a correlation between cryptanalysis and the properties of the BFs used. There is no established set of criteria for determining which mix of properties is necessary in the construction of a cryptographic BF, but some are more important than others. As various people have shown, the desired cryptographic properties of a BF generally depend on which type of cryptanalytic attack they are to withstand and the structure of the algorithm itself.

4.3.1 Balance

Perhaps the easiest property for a BF to satisfy is *balance*. A BF is *balanced* if its output is equally distributed [46]. In other words, a balanced BF on n variables has weight $wt(f) = 2^{n-1}$. In a truth table, balance is the property that half the output bits are 1 and the other half are 0. In this respect, the question of balance is a binary yes or no decision. By using this property, it can be difficult for an adversary to obtain statistical dependencies between the plaintext and ciphertext pairs [40].

4.3.2 Nonlinearity

> *Linearity is a cryptographer's worst nightmare.*
>
> $\sim$ Pante Stănică, Naval Postgraduate School (NPS) Professor

In Subsection 3.4.3, we introduced *nonlinearity* as a design criteria for the DES S-Boxes. It is not surprising that many researchers and experts feel that nonlinearity is the most important criteria for a BF to satisfy. The linear cryptanalytic attack takes advantage of linear equation schemes to break a cipher, important because linear equations can be solved in polynomial time. While it is not the aim of this thesis to describe or examine how to construct strong nonlinear BFs, the reader can delve more into this topic in [40,45,47–51].

In terms of characterization, a *nonlinear* BF is a non-affine function, i.e., a BF whose ANF contains at least one term with algebraic degree greater than one [51]. With respect to a specific function, **nonlinearity**, $\mathcal{N}_f$, is defined as the minimum Hamming distance to the

class of all affine functions, or the distance to the nearest affine function on $\mathbb{F}_2^n$ [45, 46]. Since nonlinearity is an integer valued property, functions can have varying measures of cryptographic strength. In general, a BF used for cryptography should have the highest nonlinearity possible. Of course, the nonlinearity of f is bounded above [40,45] so that the highest possible nonlinearity is

$$\mathscr{N}_f \leq 2^{n-1} - 2^{\frac{n}{2}-1}.$$

Willi Meier and Othmar Staffelbach [51] further clarified that a cryptographically good nonlinear function also needs to be "invariant under a certain group of transformations." In their example, a BF $f(x_1, x_2, \ldots, x_n)$ might contain all nonlinear terms, but a simple complement operation turns the function into a monomial with just one term. This new function under transformation is poor with respect to the number of nonlinear terms. Thus, BFs must have a large Hamming distance to the class of all affine functions to provide confusion in an algorithm [40]. Mathematically, nonlinearity is defined as

$$\mathscr{N}_f = \min_{\ell \in \mathscr{A}_n} d(f, \ell), \tag{4.9}$$

where $d(f,\ell)$ is the Hamming distance between f and an affine function ℓ, and $\mathscr{A}_n$ is the class of all affine functions on $\mathbb{F}_2^n$. The exact nonlinearity value of a BF f is given in terms of the *Walsh Transform*, which will be further explained in Section 4.5.

4.3.3 Correlation Immunity

The notion of *correlation immunity* was developed in 1984 by Thomas Siegenthaler [52], when he noted that certain stream ciphers were vulnerable to correlation attacks. Recall that in a stream cipher, the encryption scheme enciphers plaintext characters individually. As a plaintext bit moves through the cipher, a key combines with the bit to form the corresponding ciphertext. Each of these plaintext characters passing through the cipher require a key, but the process for generating the set of keys (key stream) is different for every cipher. Many stream ciphers use the LFSR technique for key stream generation. In this method, multiple LFSRs are set in parallel, with their outputs combined via a nonlinear BF to break up the linearity. The resulting combination forms the key stream. In a correlation

attack, the adversary observes a correlation between the individual LFSR outputs and the key stream [9, 53].

Thus, a BF is *correlation immune of order k* if its output is statistically independent of the combination of any k of its inputs [46]. Alternately, a BF f in n variables is correlation immune of order k, $1 \leq k \leq n$, if $P[(x(i_1), x(i_2), \ldots, x(i_k)) | f(\boldsymbol{x}) = p] = \frac{1}{2^k}$, where $x(i_i)$ is the value of the i-th bit, $p \in \mathbb{F}_2$, and P is the conditional probability of an event A given event B.

EXAMPLE 4.3.1. Consider the following truth table for a function $f(x_1, x_2, x_3)$. To check that this function is correlation immune of order 1, we must check all 1-variable subsets with their possible values and ensure that the outputs are independent of the differing inputs. The case where $f = 0$ should also be checked, but the result is the same; $P = \frac{1}{2^1} = \frac{1}{2}$.

x_3	x_2	x_1	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$
\begin{aligned}
P[x_1 = 0 | f = 1] &= 2/4 \\
P[x_1 = 1 | f = 1] &= 2/4 \\
P[x_2 = 0 | f = 1] &= 2/4 \\
P[x_2 = 1 | f = 1] &= 2/4 \\
P[x_3 = 0 | f = 1] &= 2/4 \\
P[x_3 = 1 | f = 1] &= 2/4
\end{aligned}
$$

Table 4.6: A 3-Variable BF, Correlation Immune of Order $k = 1$.

4.3.4 Resiliency

A year after Siegenthaler's introduction of correlation immunity, Benny Chor et al. introduced the term *resiliency* [54]. In [54], the authors describe a function f to be *t-resilient* if for every subset T of n input variables of cardinality t, f is unbiased with respect to T, i.e., f as a random variable is unbiased. In simpler fashion, a BF is *k-resilient* if it is both balanced and correlation immune of order k [39].

Siegenthaler was nevertheless influential in explaining how resiliency relates to correlation attacks. If a function is not k-resilient, then a correlation can be found between the

output bits and at most k input bits [40]. There is an obvious connection here with the algebraic degree of a BF. Due to Siegenthaler, we know that for a function in n variables of degree d, and correlation immune of order k, the following inequality holds: $k+d \leq n$ [52]. Furthermore, we also know that if the function is balanced and $k < n-1$, then $k+d \leq n-1 \implies d \leq n-k-1$. In cryptography, we aim to make the resiliency as high as possible. Resiliency, along with several of these other properties, can also be described in terms of the Walsh Transform (see Section 4.5).

4.3.5 Algebraic Immunity

The concept of *algebraic immunity* also arose from the study of LFSR based stream ciphers vulnerable to correlation attacks. Nicolas Courtois [55] first proposed *algebraic attacks* on these stream ciphers that either had a low-degree BF combiner or that the BF could be approximated with a low-degree polynomial. Courtois and Meier [56] later proved that this type of attack could be applied by multiplying a high-degree combiner with a carefully chosen low degree multivariate polynomial. The idea behind an algebraic attack rests on the fact that an adversary has access to some plaintext and corresponding ciphertext bits, as well as some bits of the key stream. Since the key stream is a result of the combining function, this is not too wild of an assumption. The adversary then deduces a series of low degree multivariate polynomials from each of the combiner output states, for which the key bits are solutions to. The resulting system of multivariate low degree polynomials can be solved efficiently and the secret key can be recovered [42, 53, 55, 56].

A nonzero polynomial g is called an *annihilator* of a polynomial f assuming $fg = 0$. With respect to the preceding paragraph, an annihilator of low degree aids in the implementation of an algebraic attack. Similarly, we need to consider multiples of f, i.e., $f \oplus 1$, since low degree annihilators of $f \oplus 1$ also give way to algebraic attacks [39, 40]. Thus, the *algebraic immunity* of f, denoted by $AI(f)$, is the minimum degree of g such that g is an annihilator of f or $f \oplus 1$, i.e., $AI(f) = min\{deg(g) : fg = 0 \text{ or } (f \oplus 1)g = 0\}$.

EXAMPLE 4.3.2. Given $f(x_1, x_2, x_3, x_4) = x_1x_2x_3x_4$ and $g(x_1, x_2, x_3, x_4) = x_1 \oplus x_2 \oplus x_3 \oplus x_4$, the algebraic immunity of f is 1, $AI(f) = 1$. Since $fg = x_1x_2x_3x_4 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_3x_4 = 0$, the minimum degree of g to satisfy this equation is 1, after [53].

4.3.6 Strict Avalanche Criteria and Propagation Criteria

Recall that in the explanation of the DES round function, we mentioned the notion of an *avalanche effect*. Feistel [57] was the first to use this term with regards to error detection in codes. He noted that a single error in plaintext could cause an *avalanche* of errors in the rest of the message when encrypted with a computer. Today, the avalanche effect is observed if a small change in function input yields a large change in function output [39]. With respect to a BF, the avalanche effect is present if, on average, half of the output bits change when one bit in the input is complemented (i.e., $\oplus 1$) [58].

The *strict avalanche criteria (SAC)* is an extension of the avalanche effect, requiring that "each output bit should change with a probability of one half whenever a single output bit is complemented" [58]. Formally, A. F. Webster and Tavares defined SAC in a more precise manner.

Definition 4.3.3. Let X and X_i be n-bit binary plaintext vectors, such that X and X_i differ in one bit, $1 \leq i \leq n$, i.e., $wt(X \oplus X_i) = 1$. Let $V_i = Y \oplus Y_i$, where $Y = f(X), Y_i = f(X_i)$ and f is a function. If f satisfies the SAC, then the probability that each bit in V_i is equal to one should be one half over the set of all possible plaintext vectors X and X_i.

Kwangjo Kim and others [49,59,60] provide a more implementable definition of SAC. Let $\boldsymbol{c}_i^{(n)}$ denote an n dimensional vector with Hamming weight one at the i-th position.

Definition 4.3.4. A function $f : \mathbb{F}_2^n \to \mathbb{F}_2^m$ satisfies the SAC if for all i $(1 \leq i \leq n)$ the following equations hold:

$$\sum_{\boldsymbol{x} \in \mathbb{F}_2^n} \left(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus \boldsymbol{c}_i^{(n)}) \right) = (2^{n-1}, 2^{n-1}, \ldots, 2^{n-1}). \tag{4.10}$$

Definition 4.3.4 is the most general definition for any function, but since we are mainly concerned with BFs, the codomain is just $\mathbb{F}_2$ and the right hand side of the equation is just 2^{n-1}. Thus, a one bit change in the 2^n input vectors results in an output change for 2^{n-1} of those vectors (i.e., exactly half). Example 4.3.5 demonstrates the SAC for a BF on three variables.

EXAMPLE 4.3.5. In this BF with $n = 3$, the possible one-bit changes are reflected to

the right of the original function output column. Note that for each bit change, the output changes for exactly $2^2 = 4$ vectors.

x_3	x_2	x_1	f	$\oplus 100$	$\oplus 010$	$\oplus 001$
0	0	0	1	0	1	1
0	0	1	1	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	1	1
1	0	0	0	1	1	1
1	0	1	1	1	1	0
1	1	0	1	1	0	1
1	1	1	1	0	1	1

Table 4.7: A 3-Variable BF Satisfying the SAC, after [39].

Another result that follows from the SAC is balance in the Hamming weights between the contrasting outputs. This result is also from Webster and Tavares [58], but is formalized by Cusick and Stănică [39] as a lemma.

Lemma 4.3.6. A BF $f : \mathbb{F}_2^n \to \mathbb{F}_2$ satisfies the SAC iff the function $f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus \boldsymbol{a})$ is balanced for every $\boldsymbol{a}$ in $\mathbb{F}_2^n$ with Hamming weight 1.

As visualization of this lemma, refer back to Table 4.7. Note that the XOR between the f column and any of the bit change columns is a balanced string. Although it was developed in 1986, SAC was generalized a few years later.

In 1990, Bart Preneel et al. generalized SAC as *propagation criteria*. A BF satisfies the *propagation criteria of degree* $\boldsymbol{k}$, denoted as $PC(k)$, if $f(\boldsymbol{x})$ changes with a probability of one half whenever i ($1 \leq i \leq k$) of the n bits of $\boldsymbol{x}$ are complemented [61]. Given this definition, SAC is equivalent to $PC(1)$.

Just like with SAC, there are alternate ways to present the definition of PC. One such definition relies on the concept of a *directional derivative* of a BF. If f is a BF in n variables and $\boldsymbol{b}$ is any vector in $\mathbb{F}_2^n$, then the *derivative of f in the direction of* $\boldsymbol{b}$ is

$D_{\boldsymbol{b}}f(\boldsymbol{x}) = f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus \boldsymbol{b})$ [40]. Hence, a BF $f(\boldsymbol{x})$ in n variables satisfies $PC(k)$ if and only if all of the directional derivatives are balanced functions, i.e., for all $\boldsymbol{a} \in E \subset \mathbb{F}_2^n$, the derivative $D_{\boldsymbol{a}}f(\boldsymbol{x}) = f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus \boldsymbol{a})$ is balanced [39, 40].

4.3.7 Other Properties

There are other criteria for BFs that are not as prevalent in mainstream literature, but have gained notoriety in recent research. We start with two properties that have either already been defined or do not require definition. The first of these is the aforementionied *algebraic degree*. The algebraic degree contributes to the complexity of a BF and is often a factor in attacks on ciphers; we typically want to employ BFs with the highest algebraic degree possible. Algebraic attacks are very efficient against ciphers employing low degree polynomials [42], and the complexity of the differential attack of higher order depends on the highest degree of the BF used in the cryptosystem [45, 62].

Just because a BF has high degree, however, does not make it cryptographically relevant. We saw in Subsection 4.3.2 that via a complement operation, a function was transformed into a monomial. Even though this monomial might have high algebraic degree, it is weak when compared to a polynomial of same degree. Thus, the other property we consider is the number of terms in the ANF. The BFs that were discussed in Subsection 4.3.3 as nonlinear combining functions in stream based LFSRs need to have high algebraic degree and many terms in the ANF in order to resist key stream generation by the Berlekamp-Massey Algorithm [45]. The number of terms in the ANF is not a stand alone property however. Along with the same reasoning just presented, a BF with many terms could have an affine equivalent function under a transformation. Thus, this property needs to be considered with other properties, such as affine invariance, algebraic degree, etc.

Motivated by the work of Meier and Staffelbach [51], Carlet introduced a new property with respect to the number of terms in the ANF, i.e., an affine invariant parameter. Carlet called this property the *algebraic thickness* of a BF. The algebraic thickness, denoted by $\mathscr{T}(f)$, is defined to be the minimum number of terms in the ANF of the set of functions $f \circ A$, where A is the general affine group, and A ranges over the set of all affine automorphisms of $\mathbb{F}_2^n$ [45, 63, 64]. As Carlet points out, we would like to work with BFs having the highest possible algebraic thickness, but "classical BFs have small algebraic thickness" [45]. Carlet

is not explicit in what he denotes as *classic*, though one can infer that he means those BFs we are most interested in with respect to cryptographic applications. The algebraic thickness is bounded by the number of variables in the polynomial, i.e., 2^n, but it is unproven that there exist functions f for which $\mathscr{T}(f) > 2^{n-1}$ [45].

There are still other parameters that exist for which the interested reader should consult the references. One such example is the global avalanche criteria (GAC) as presented by Xian-Mo Zhang and Yuliang Zheng [65]. Both SAC and PC are known to be *local* characteristics of a function, namely that they guarantee avalanche features for vectors of Hamming weight either 1 or up to k. SAC and PC are restrictive, however, because they can admit functions having a large Hamming weight with vectors as linear structures. SAC also requires that $f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus \boldsymbol{a})$ is balanced, which rules out *bent functions* (see next section). Other properties include *maximum correlation* [40, 66], *nonhomomorphicity* [40, 67], and *non-k-normality* [40].

4.4 Bent Boolean Functions

We have mentioned *bent functions* several times, and now a short background is presented. Since bent BFs are not the focus of this thesis, the reader should consult the works of John Dillon, Oscar Rothaus, Robert McFarland, W. Meier, and others [39, 51, 68–70] for more on this subject.

Bent BFs are desirable in cryptography because they achieve the maximum nonlinearity for a BF, but they are difficult to implement. One such reason was mentioned in the previous section—bent functions have desirable properties, but they are not balanced, and we want balanced functions as S-Boxes.

Definition 4.4.1. A BF f on $\mathbb{F}_2^n$ is called **bent** if its Hamming distance to the set of all n-variable affine functions equals $2^{n-1} - 2^{\frac{n}{2}-1}$. In other words, a **bent** function achieves the maximum possible nonlinearity, $\mathscr{N}_f$, for any BF in n variables. Furthermore, this distance is only achieved when n is even [40, 45].

As a result of the definition, bent functions also achieve many other characteristics. If an n-variable BF is bent with n even, then it satisfies $PC(n)$ [39, 40]. Meier and Staffelbach's

perfect nonlinear functions are essentially an analagous form of bent functions [51]. There is also a definition of bent functions that uses the Walsh transform (see next section).

Although it seems that bent functions are desirable and we should be using them, the mystery surrounding them lies in construction. We know the total number of bent functions for $n = 2, 4, 6, 8$ variables, but we do not know the total for $n \geq 10$. Thus, we have no means to characterize or classify this set of bent functions under the general affine group [40]. The main difficulty here lies in the space of possible bent functions. For $n = 2$, there are 16 possible BFs and eight total bent functions. Remarkably, for $n = 8$, there are 2^{256} BFs and approximately $2^{106.291}$ total bent functions [39, 71].

4.5 Walsh Transform

Most readers are familiar with the concept of a mathematical transform. A transform is a relation that takes a function in one domain or basis and *transforms* it into a function in another domain or basis. A classic example of this is the *Laplace Transform*, which takes a function $f(t)$ and outputs a new function $F(s)$. We now examine another famous transform, the *Fourier Transform*, which allows a transfer between the time (or spatial) domain and the frequency domain.

The Fourier Transform has many applications, some of which include acoustics, digital signal processing, physics, engineering, and image processing. It is essentially an extension of the Fourier series, in which periodic behavior is modeled by an infinite sum of sines and cosines. We are interested in the non-continuous version of the Fourier Transform called the *discrete Fourier Transform (DFT)*. In the DFT, the function used as the input is discrete and its values are given over a finite interval. This transform is also invertible so that we can move back and forth between bases.

With regard to BFs, the DFT is an invertible mapping of the function values onto a set of coefficients, called Fourier coefficients [72]. Knowledge of the Fourier coefficients gives information about the function, such as computational complexity and other properties of BFs. In particular, the DFT of a function gives the weights of all functions of the form $f \oplus \ell$, where ℓ is affine [40]. The DFT of BFs is also called the *Walsh Transform (WT)*.

Recall from linear algebra that a basis for a vector space is a set of linearly independent

vectors that can span that space, i.e., every vector in the vector space can be represented by a linear combination of the basis vectors. By doing so, we find the coordinates of every point in the space with respect to that basis. This can be difficult if the basis vectors are not orthogonal. If we can find an *orthogonal* basis for the vector space, then we can define an inner (dot) product and expressing all vectors in the vector space is much easier.

In the most general sense, a BF is a 0-1 valued real function defined on $\{0,1\}^n$, i.e., $f : \mathbb{F}_2^n \to \mathbb{R}$. If we restrict the codomain of f to only the two-valued functions on this domain, then we consider $f : \mathbb{F}_2^n \to \mathbb{F}_2$. The domain of the space of all these functions is an Abelian group, for which we define a *group character*, $Q_{\boldsymbol{w}}(\boldsymbol{x}) = (-1)^{<\boldsymbol{w}\cdot\boldsymbol{x}>}$. The notation $<\boldsymbol{w}\cdot\boldsymbol{x}>$ is the inner (dot) product on vectors over $\mathbb{F}_2$, $w_1x_1 \oplus w_2x_2 \oplus \cdots \oplus w_nx_n$. The set of functions $\{Q_{\boldsymbol{w}} : \boldsymbol{w} \in \mathbb{F}_2^n\}$ forms an orthogonal basis for the vector space $\mathbb{F}_2^n$ [72]. The WT then defines the coefficients of the BF f with respect to this orthogonal basis.

Definition 4.5.1. [39, 73] If f is any real-valued function on $\mathbb{F}_2^n$, i.e., $f : \mathbb{F}_2^n \to \mathbb{R}$, then the **Walsh Transform (WT)**[4] of f on a vector $\boldsymbol{w}$ is defined by

$$F(\boldsymbol{w}) = W(f)(\boldsymbol{w}) = \sum_{\boldsymbol{x}\in\mathbb{F}_2^n} f(\boldsymbol{x}) \cdot (-1)^{<\boldsymbol{w}\cdot\boldsymbol{x}>}, \tag{4.11}$$

where $\boldsymbol{w} \in \mathbb{F}_2^n$ and $<\boldsymbol{w}\cdot\boldsymbol{x}>= w_1x_1 \oplus w_2x_2 \oplus \cdots \oplus w_nx_n$ over $\mathbb{F}_2$. The function f can be recovered from $F(\boldsymbol{w})$ by the **inverse Walsh Transform**

$$f(\boldsymbol{x}) = W^{-1}(F)(\boldsymbol{x}) = 2^{-n} \sum_{\boldsymbol{w}\in\mathbb{F}_2^n} F(\boldsymbol{w}) \cdot (-1)^{<\boldsymbol{w}\cdot\boldsymbol{x}>}. \tag{4.12}$$

Of course, the BF f takes on the real values $\{0,1\}$, but sometimes it is easier to work with BFs that take on values in the range $\{-1,1\}$. This alternate group of functions will be denoted by $\hat{f}$. The function $\hat{f}$ is related to the function f in the following manner

$$\hat{f}(\boldsymbol{x}) = (-1)^{f(\boldsymbol{x})} \qquad \text{or} \qquad \hat{f}(\boldsymbol{x}) = 1 - 2f(\boldsymbol{x}). \tag{4.13}$$

[4]We acknowledge that the nomenclature within the Walsh Transform is varied. Some sources call this definition the *Hadamard Transform*, the *discrete Fourier-Walsh-Hadamard Transform*, or the *Walsh-Hadamard Transform*. Unfortunately, there is no standard definition, but the notation presented here is adopted from [39, 73].

The function on the left in Equation 4.13 is often referred to as the *sign function*, for which the WT also exists. This transform, however, we will call the Walsh-Hadamard Transform (WHT).

Definition 4.5.2. The **Walsh-Hadamard Transform** of $\hat{f}$ is given by

$$\hat{F}(\boldsymbol{w}) = W(\hat{f})(\boldsymbol{w}) = \sum_{\boldsymbol{x}\in\mathbb{F}_2^n} (-1)^{f(\boldsymbol{x})\oplus<\boldsymbol{w}\cdot\boldsymbol{x}>}. \tag{4.14}$$

In the same way that f and $\hat{f}$ are related, there is also a relationship between the WT and the WHT. This is a rather important relationship, thus it is stated as a lemma. The simple proof is omitted, but is available in [39].

Lemma 4.5.3. If $\hat{f}(\boldsymbol{x}) = (-1)^{f(\boldsymbol{x})}$, then

$$\hat{F}(\boldsymbol{w}) = -2F(\boldsymbol{w}) + 2^n\delta(\boldsymbol{w}), \tag{4.15}$$

or

$$F(\boldsymbol{w}) = 2^{n-1}\delta(\boldsymbol{w}) - \frac{1}{2}\hat{F}(\boldsymbol{w}), \tag{4.16}$$

where $\delta(\boldsymbol{w})$ is the *Kronecker delta* function (sometimes called the *Dirac symbol*) defined as

$$\delta(\boldsymbol{w}) = \begin{cases} 1, & \text{if } \boldsymbol{w} = \boldsymbol{0} \\ 0, & \text{otherwise.} \end{cases}$$

Equations 4.11 and 4.14 each yield a vector of Fourier coefficients as $\boldsymbol{w}$ varies, also known as *Walsh coefficients*. These lists of 2^n coefficients are called the *Walsh spectrum* of f and the *Walsh-Hadamard spectrum* of $\hat{f}$, respectively [39]. For general purposes, we refer to either list as the Walsh spectrum of a BF, although context should be clear upon which version is presented. The Walsh spectrum is another unique representation of a BF and is often used as a means to explicitly define certain cryptographic properties on a function. We will return to this notion shortly, but first we present an example of the WT.

EXAMPLE 4.5.4. Both the WT and WHT involve sums over the entire vector space $\mathbb{F}_2^n$. Thus, by-hand calculations are rarely practical. Consider the BF defined as $f : \mathbb{F}_2^2 \to \mathbb{F}_2$, with ANF given by $1 \oplus x_1 \oplus x_2$. The truth table representation is given in Table 4.8.

x_2	x_1	f
0	0	1
0	1	0
1	0	0
1	1	1

Table 4.8: Truth Table Representation for $1 \oplus x_1 \oplus x_2$.

WT

$$F(\boldsymbol{w}) = W(f)(\boldsymbol{w}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^2} f(\boldsymbol{x}) \cdot (-1)^{<\boldsymbol{w} \cdot \boldsymbol{x}>}$$

$$F(00) = 1(-1)^0 + 0 + 0 + 1(-1)^0 = 2$$
$$F(01) = 1(-1)^0 + 0 + 0 + 1(-1)^1 = 0$$
$$F(10) = 1(-1)^0 + 0 + 0 + 1(-1)^1 = 0$$
$$F(11) = 1(-1)^0 + 0 + 0 + 1(-1)^2 = 2$$

Walsh spectrum = (2,0,0,2)

WHT

$$\hat{F}(\boldsymbol{w}) = W(\hat{f})(\boldsymbol{w}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{f(\boldsymbol{x}) \oplus <\boldsymbol{w} \cdot \boldsymbol{x}>}$$

$$\hat{F}(00) = (-1)^{1 \oplus 0} + (-1)^{0 \oplus 0} + (-1)^{0 \oplus 0} + (-1)^{1 \oplus 0} = 0$$
$$\hat{F}(01) = (-1)^1 + (-1)^1 + (-1)^0 + (-1)^0 = 0$$
$$\hat{F}(10) = (-1)^1 + (-1)^0 + (-1)^1 + (-1)^0 = 0$$
$$\hat{F}(11) = (-1)^1 + (-1)^1 + (-1)^1 + (-1)^1 = -4$$

Walsh-Hadamard spectrum = (0,0,0,-4)

The reader can easily verify the relation between the two spectra using Equation 4.15 and that the truth table output can be recovered by the inverse in Equation 4.12. Note that the Kronecker delta function is only equal to one when $\boldsymbol{w}$ is the zero vector.

Since the WT operates as a DFT, the classical method of solving for the Fourier coefficients is not an integral problem but rather a matrix problem. Thus, the Walsh spectrum can also be found by means of *Hadamard matrices*. Hadamard matrices are recursively constructed and consist of ± 1s. Formally [39], a Hadamard matrix H of order n is an $n \times n$ matrix of ± 1s such that $HH^T = nI_n$, where H^T is the transpose of H and I_n is the $n \times n$ identity matrix. The recursion is given as

$$H_0 = [1]; \qquad H_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad \text{and} \qquad H_n = \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix}. \tag{4.17}$$

Thus, H_2 is constructed in typical block matrix style as

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}.$$

Therefore, expressed as a matrix product, the WT is given by [46, 61]

$$[F] = H_n \cdot [f], \tag{4.18}$$

where $[F]$ is a column vector of the Walsh spectrum values and $[f]$ is a column vector of the function values. Returning to Example 4.5.4, we can compute the Walsh spectrum using the Hadamard matrix, but again note for large values of n, computations by-hand become impractical quickly.

$$[F] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 0 \\ 2 \end{bmatrix}$$

Similarly, the WHT can be expressed in terms of the Hadamard matrix as $[\hat{F}] = H_n \cdot [(-1)^f]$,

where $[\hat{F}]$ is a column vector of the Walsh-Hadamard spectrum and $[(-1)^f]$ is a column vector of negative ones raised to the function values [46].

$$[\hat{F}] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -4 \end{bmatrix}$$

We now return to the concept alluded to in the previous section concerning the WT and cryptographic properties of BFs. There are a number of properties related to the WT/WHT, namely because the transform is a linear mapping and provides information on nonlinearity [46, 72]. We must be careful to define which transform is being used though, which should be clear in the notation. Other properties, such as SAC and PC, are related to the *autocorrelation function*, which we do not discuss here but can be found in [39].

Balance: [46] A BF is balanced if $\hat{F}(\mathbf{0}) = 0$. This feature is observed in Example 4.5.4.

Nonlinearity: The nonlinearity of f is determined by the WHT of $\hat{f}$ [39], that is,

$$\mathcal{N}_f = 2^{n-1} - \frac{1}{2} \max_{\boldsymbol{u} \in \mathbb{F}_2^n} |\hat{F}(\boldsymbol{u})|, \tag{4.19}$$

where the bars represent absolute value. The function in Example 4.5.4 has nonlinearity zero since $2^1 - \frac{1}{2}(4) = 0$.

Correlation Immunity: [39] A BF is correlation immune of order k, $1 \leq k \leq n$, if and only if $\hat{F}(\boldsymbol{w}) = 0$ for $1 \leq wt(\boldsymbol{w}) \leq k$. The function in Example 4.5.4 is correlation immune of order one since both $\hat{F}(01) = 0$ and $\hat{F}(10) = 0$.

Resiliency: [46] Since resiliency also includes correlation immunity, the same stipulations on the WHT apply here. Thus, the resiliency for the function in Example 4.5.4 is also one.

Bent BFs: A BF in n variables is bent if and only if $\hat{F}(\boldsymbol{u}) = \pm 2^{n/2}$ for all $\boldsymbol{u} \in \mathbb{F}_2^n$ [51, 68]. The function $f(\boldsymbol{x}) = x_1 x_2$ on $\mathbb{F}_2^2$ is bent since the Walsh-Hadamard spectrum is $|\hat{F}(\boldsymbol{u})| = 2^{2/2} = (2, 2, 2, -2)$. Another version of Fourier spectrum is the *energy spectrum*. The energy spectrum is defined as the square modulus of the Fourier transform [61], i.e., $\hat{F}^2$. In this

manner, all coefficients are positive constants. With respect to the energy spectrum of a BF, we often characterize a bent function as having a flat spectrum.

4.6 Vectorial Boolean Functions

Recall that an S-Box is a mapping or substitution from an m-bit input to an n-bit output, where m and n need not be equal. Over the binary field, this is represented by $f : \mathbb{F}_2^m \to \mathbb{F}_2^n$. These functions are also called (m,n)-functions, *multi-output BFs*, *vectorial BFs*, and *S-Box*es [48]. Vectorial BFs employed in iterative block ciphers are used to provide confusion in the algorithm. Much work in the area of vectorial BFs for cryptography has been done by Carlet [40, 48].

Given that m and n are positive integers, if a function F exists as an (m,n)-function, then the BFs $f_1, f_2, \ldots, f_n$ defined at every $\boldsymbol{x} \in \mathbb{F}_2^m$ by $F(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x}))$ are called the *coordinate functions* of F [48]. In the case of DES, each of the eight S-Boxes are functions $f : \mathbb{F}_2^6 \to \mathbb{F}_2^4$. Within each S-Box, we treat the four rows as coordinate functions. Thus, for any S-Box, there exists $F(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), f_3(\boldsymbol{x}), f_4(\boldsymbol{x}))$, where each f_i is a mapping from $\mathbb{F}_2^6$ to $\mathbb{F}_2$. Our aim in this thesis is to examine the coordinate functions of the S-Boxes.

There has been extensive research on the construction of cryptographically *good* S-Boxes. The DES creators stated that the boxes were built to resist a differential attack. One such method for doing so requires that the output of an (m,n)-function F to its derivatives $D_a(\boldsymbol{x}) = F(\boldsymbol{x}) + F(\boldsymbol{x} + \boldsymbol{a})$ must be distributed as uniformly as possible [48]. There is also a method for designing against Matsui's linear attack, which deals with linear combinations of the coordinate functions [48].

The DES S-Boxes have received much attention over the years. Webster, Tavares, and Adams, while writing in terms of generic S-Boxes, have always used DES as influence in their analysis. For example, in [58], the authors show that the set of DES S-Boxes do not satisfy the SAC; the probability that an output bit will change when a single input bit is complemented varies from 0.43 to 0.93. Granted, SAC did not exist at the time when IBM created DES. S-Box construction has also been studied from the viewpoints of random generation versus systematic design. While random generation is often effective, the design criteria mentioned by Adams and Tavares [50] is worth noting.

According to Adams and Tavares, an S-Box must satisfy the following criteria to be "cryptographically desirable":

1. bijection;
2. nonlinearity;
3. strict avalanche;
4. independence of output bits.

Property (1) observes that a $2^n \times n$ S-Box is bijective, i.e., invertible (which may or may not be necessary). In doing so, the input vectors map to distinct output vectors and the output vectors appear only once per stage. Property (2) is obvious, but in order to ensure nonlinearity at both the bit level and integer level, the S-Box must utilize n nonlinear BFs. As a consequence of Property (1), Property (2) is typically achieved in the inverse S-Box. Property (3) was introduced in [58], but an S-Box as a whole possesses the SAC if it has Properties (1) and (4), and all n BFs fulfill the SAC. To show this, Adams and Tavares used Forré's method of construction for SAC-fulfilling BFs [73]. Property (4) is intended to resist certain correlation attacks. Others such as K. Kim have done more recent research into the construction of good S-Boxes; for a survey of these techniques, consult [47, 49, 59, 60].

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Basic Graph Theory

Graph theory is the study of *graphs*, but not the typical function graph depicted on say the $x-y$ plane. Instead, graph theory examines the relations between objects, be them people, places, devices, molecules, etc. Since the field implicates models of everyday life, some refer to graphs as networks. Most scholars date the origin of graph theory to the famous Königsberg bridge problem solved by Euler in 1736. While it is a fairly old discipline, tremendous advances in graph theory, especially regarding networks, have spurned interest in the field within the last century. There are many terms within graph theory that are not defined here, but the reader can consult a standard graph theory text such as [74] for more insight.

5.1 Definitions

A graph is a collection of objects called *vertices* and the relations between them called *edges*. Sometimes, vertices are also called *nodes* while edges are also called *arcs*.

Definition 5.1.1. [74] A **graph** G is an ordered pair (V,E), where V is the finite set of vertices of G and E is the set of two-element subsets of V called edges. V is called the **vertex set** of G and E is called the **edge set** of G. The cardinality of V is called the **order** of the graph G, denoted by n.

A graph can be uniquely represented by the ordered pair (V,E) or by a pictorial model. Consider Example 5.1.2 where this is depicted.

EXAMPLE 5.1.2. In Figure 5.1, G is given by (V,E), where $V=\{v_1,v_2,v_3,v_4,v_5\}$ and $E=\Big\{\{v_1,v_2\},\{v_2,v_3\},\{v_3,v_4\},\{v_4,v_5\},\{v_1,v_5\},\{v_1,v_3\},\{v_1,v_4\}\Big\}$. Ordinarily, we omit the set notation on the vertex pairs, so E can be written as
$E=\{v_1v_2,\ v_2v_3,\ v_3v_4,\ v_4v_5,\ v_1v_5,\ v_1v_3,\ v_1v_4\}$. This graph is *undirected*, in that there is no orientation on the edges. This graph is also *simple* because there are no *loops* or *multiple edges*.

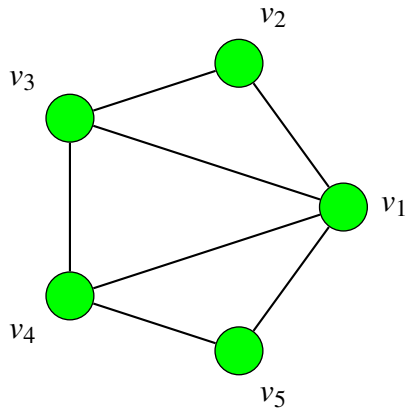


Figure 5.1: A Graph G on $n = 5$ Vertices.

Note in Example 5.1.2 that if the edge v_iv_j is in the edge set of G, then it appears as a line segment (or curve) connecting vertex v_i with vertex v_j. If the edge v_iv_j exists, i.e., $v_iv_j \in E(G)$, then we say that v_i and v_j are *adjacent*. If v_i and v_j are adjacent, then they are also referred to as *neighbors*. If an edge e joins vertices v_i and v_j, then we say that e is *incident* with v_i (as well as v_j).

Some graphs allow for multiple connections between two vertices. For example, an airline might plan several routes between Detroit and San Francisco, depending on weather, traffic, or other variables. In this case, the airline route graph can depict multiple routes, which we call a *multigraph*. If an edge is also permitted to join a vertex to itself, then this graph is called a *pseudograph*. Figure 5.2 depicts these types of graphs.

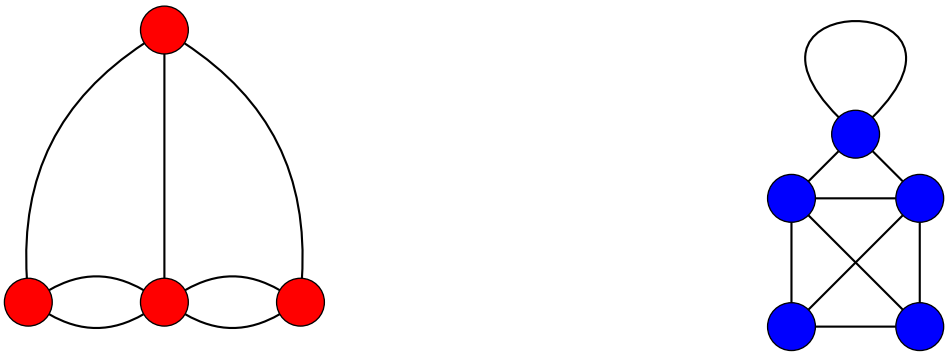


Figure 5.2: Multigraph and Pseudograph, Respectively.

A multigraph contains at least one pair of distinct vertices that are joined by multiple (parallel) edges. Multigraphs do not permit loops. A pseudograph permits multiple edges and loops, but does not necessarily contain multiple edges. In this thesis we will consider simple graphs and pseudographs.

The *degree of a vertex* can be defined in two synonymous ways. The degree of $v \in V(G)$ is equal to the number of edges incident with v. We also have that the degree of $v \in V(G)$ is the number of vertices adjacent to v [74]. The degrees of the vertices within the graphs of Figure 5.2 can be represented as sequences: $(3,3,3,5)$ and $(3,3,4,4,4)$[5], respectively. There are various rules, theorems, and bounds pertaining to vertex degree, but again we assume that the reader has knowledge of these or can consult a standard reference.

Additionally, a graph G is *regular* if all vertices of G have the same degree. A graph G is *r-regular* if $deg(v) = r$ for all $v \in V(G)$.

5.2 Matrix Representations

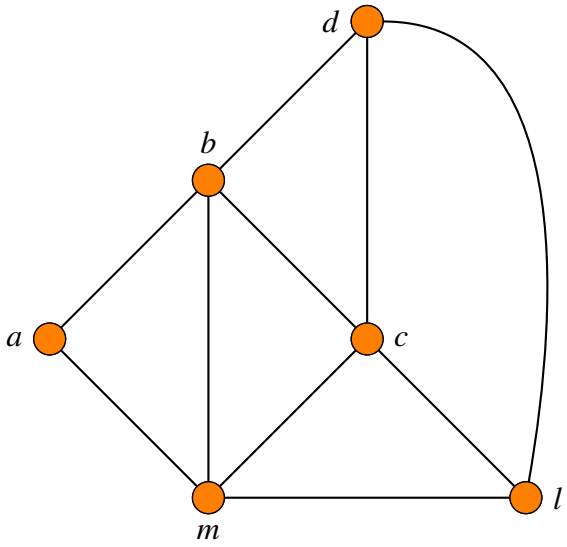
A graph can also be represented by a matrix describing the relations on vertices and edges. The most widely used matrix to describe a graph is the *adjacency matrix*. Like the name implies, the adjacency matrix displays the vertex adjacencies of the edge set of G (as well as the non-adjacencies).

Definition 5.2.1. [74] Assume that G is a simple, undirected graph of order n with vertex set $\{v_1, v_2, \ldots, v_n\}$. The **adjacency matrix** of G is the $n \times n$ matrix $\boldsymbol{A} = [a_{ij}]$, whose entries a_{ij} are given by

$$a_{ij} = \begin{cases} 1, & \text{if } v_i v_j \in E(G) \\ 0, & \text{otherwise.} \end{cases}$$

Figure 5.3 illustrates the concept of an adjacency matrix. The labeling of vertices outside the adjacency matrix is not a common practice, but this is displayed for the benefit of the reader.

[5]Note that for the loop, we counted the degree twice for the loop. While some graph theorists and authors only consider a loop to contribute one towards the vertex degree, the majority of texts double count the degree for a loop.



$$\boldsymbol{A} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ l \\ m \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & l & m \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

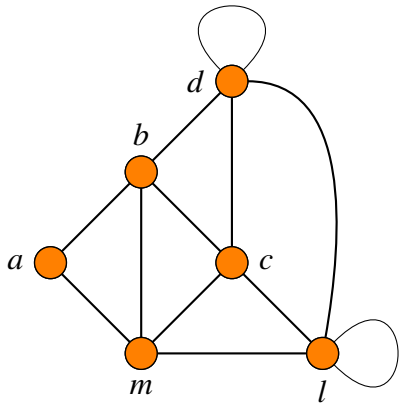
Figure 5.3: A Graph and Its Associated Symmetric Adjacency Matrix.

There are a couple of observations [75, 76] to make with respect to the adjacency matrix for a simple, undirected graph.

i) $\boldsymbol{A}$ is a real and symmetric matrix;
ii) The row sums for each i of $\boldsymbol{A}$ equal the degree of each v_i;
iii) The diagonal entries of $\boldsymbol{A}$ are zero;
iv) The *trace* of $\boldsymbol{A}$ is zero, i.e., $tr(\boldsymbol{A}) = \sum_{i=1}^{n} a_{ii} = 0$;
v) There is a one-to-one correspondence between the graph G and its associated adjacency matrix $\boldsymbol{A}$ (up to isomorphism and rearrangement of vertices in $\boldsymbol{A}$);
vi) $\boldsymbol{A}$ is not unique, since we can reorder the vertices and arrive at a different representation.

Adjacency matrices for multigraphs are formed in a similar manner, in that the entry a_{ij} is the number of edges vetween v_i and v_j. In a pseudograph, however, we must now account for loops which implies nonzero entries on the diagonal. Unfortunately, there is no standard method to handle the entry a_{ii} in the adjacency matrix of a pseudograph. Some propose that a loop should be given a *weight* of two (i.e., the entry a_{ii} is twice the number of loops attached to the vertex v_i [77]). This vertex-centric approach allows the adjacency matrix to hold the properties of row sums equaling the degree as well as the **First Theorem of Graph Theory**.[6] Others model a loop should be given a weight of one, which leans toward an

edge-centric approach [78]. For this thesis, we use the latter approach, the reasons for which will become apparent in Section 5.4. Consider Figure 5.4 as an example of our approach to pseudographs.



$$\boldsymbol{A} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ l \\ m \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & l & m \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Figure 5.4: A Pseudograph and Its Associated Adjacency Matrix.

The most common approach to multigraphs and pseudographs is to consider them as *weighted graphs*. In this respect, we assign each edge a weight. If an edge is not present, it has a weight of zero. Thus, this allows all graphs to be treated as weighted graphs, with an assigned weight function satisfying $W : V \times V \to \mathbb{R}$, with $w(i, j) = w(j, i)$ and $w(i, j) \geq 0$ [79]. The weight function W also has the properties that $w(i, j) > 0$ if and only if $ij \in E(G)$. With this application, a simple, undirected, and unweighted graph is a special case where the weights are either one or zero. Therefore, we use the terms *adjacency matrix* and *matrix of weights* interchangeably. This weighting does allow for the possibility of an adjacency matrix that is not in the traditional 0-1 format, but given our approach in Figure 5.4 we will not consider this.

Another matrix representation for a graph is the *Laplacian*. The Laplacian matrix has a long history dating back to German physicist Gustav Kirchhoff. In 1847, Kirchhoff developed the basis for the *matrix-tree theorem* (see [74]), which uses the Laplacian matrix in its construction. Therefore, the Laplacian is also referred to as the *Kirchhoff matrix* [80].

[6]The First Theorem of Graph Theory states that the sum of the degrees in a graph G is equal to twice the number of edges in G.

Definition 5.2.2. [79, 80] Let G be a graph, possibly weighted, of order n. The **Laplacian matrix** of G is the $n \times n$ matrix $\boldsymbol{L} = [L_{ij}]$, whose entries L_{ij} are given by

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A},$$

where $\boldsymbol{D}$ is the diagonal matrix indexed by $V(G)$, with $i, j \in V(G)$, $deg(i) = d(i) = \sum_i a_{ij} = \sum_i w(i,j)$ and $\boldsymbol{A}$ is the adjacency matrix. In an equivalent fashion,

$$\boldsymbol{L} = \begin{cases} d(i) - w(i,\ i), & \text{if } i = j \\ -w(i,\ j), & \text{if } ij \in E(G) \\ 0, & \text{otherwise.} \end{cases}$$

Unfortunately, the Laplacian does not have a one-to-one correspondence with a graph G. It is mainly used to deduce properties of the (possibly unknown) graph. However, it is a real symmetric matrix, and in fact the Laplacian is a positive semidefinite, singular matrix. Consider Example 5.2.3 in which the Laplacian is computed for the graph in Figure 5.4.

EXAMPLE 5.2.3.

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 & -1 \\ -1 & 4 & -1 & -1 & 0 & -1 \\ 0 & -1 & 4 & -1 & -1 & -1 \\ 0 & -1 & -1 & 3 & -1 & 0 \\ 0 & 0 & -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 0 & -1 & 4 \end{bmatrix}$$

There are still other matrices that can be used to represent a graph such as the *incidence matrix*, *distance matrix*, *normalized Laplacian*, *signless normalized Laplacian*, and *signless Laplacian*. However, these matrices are not the focus of this thesis.

5.3 Spectral Graph Theory

The field of linear algebra is rich with techniques for examining structural properties of matrices. With the ability to represent a graph by a matrix, these techniques now become available to the user. This field is known as *algebraic graph theory*, in which we attempt to determine properties of graphs using algebraic properties of the matrices representing them [81, 82]. *Spectral graph theory* is a subfield of algebraic graph theory which specifically aims to examine graph properties using the *spectrum* of a graph's associated matrix. The classic references on this subject are found in the works of Biggs [76], Cvetkovíc et al. [77], and Chung [79]. The importance of spectral graph theory can be observed in the following quotations.

> Just as astronomers study stellar spectra to determine the make-up of distant stars, one of the main goals in graph theory is to deduce the principal properties and structure of a graph from its graph spectrum. The spectral approach for general graphs is a step in this direction. There is no question that eigenvalues play a central role in our fundamental understanding of graphs. [79]

> Spectral graph theory is a useful subject. The founders of Google computed the Perron-Frobenius eigenvector of the web graph and became billionaires. [81]

5.3.1 Definitions

Definition 5.3.1. [76, 81] The (ordinary) **spectrum** of a finite graph G of order n is the spectrum of the adjacency matrix $\boldsymbol{A}(G)$, that is the set of n eigenvalues of $\boldsymbol{A}(G)$ together with their (algebraic) multiplicities. If the distinct eigenvalues of $\boldsymbol{A}(G)$ are $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ and their multiplicities are $m(\lambda_1), m(\lambda_2), \ldots, m(\lambda_n)$, then we shall write

$$\operatorname{Spec} G = \begin{pmatrix} \lambda_1 & \lambda_2 & \cdots & \lambda_n \\ m(\lambda_1) & m(\lambda_2) & \cdots & m(\lambda_n) \end{pmatrix}.$$

Similarly, the *Laplace spectrum* of a finite graph G is the spectrum of the Laplacian matrix $\boldsymbol{L}$ [81]. Note that Definition 5.3.1 does not include the corresponding *eigenvectors*. This is mainly due to the fact that eigenvectors are not unique, and that for a given eigenvalue λ,

any scalar multiple of a nonzero vector $\boldsymbol{x}$ satisfies the eigenvalue problem: $A\boldsymbol{x} = \lambda\boldsymbol{x}$. There are certain graph properties that do account for eigenvectors, but in general we will not be concerned with them here.

Recall that to find the n eigenvalues of an $n \times n$ matrix A, we must find the n roots of the characteristic polynomial $p(\lambda) = \det(A - \lambda I)$. Since the adjacency matrix $\boldsymbol{A}$ is real and symmetric, its eigenvalues are also real numbers. Likewise, since the Laplacian $\boldsymbol{L}$ is positive semidefinite, its eigenvalues are all nonnegative (i.e., $\lambda_i \geq 0$ for all $i \in \{1, 2, \ldots, n\}$) and a zero eigenvalue is guaranteed (since the row sums are zero) [80]. Additionally, the algebraic and geometric multiplicity of each eigenvalue is the same, hence *multiplicity* is used interchangeably.

5.3.2 Some Known Results

We now present some of the many known results on graph spectra. Some of these deal with the adjacency matrix and some deal with the Laplacian. From context it should be clear which matrix is being used. Also, it should be apparent that if λ is an eigenvalue of the adjacency matrix $\boldsymbol{A}$ for an r-regular graph G, then $r - \lambda$ is an eigenvalue of the Laplacian $\boldsymbol{L}$. For added clarity, we refer to the eigenvalues of $\boldsymbol{A}$ as $\lambda_1, \lambda_2, \ldots, \lambda_n$ and the eigenvalues of $\boldsymbol{L}$ as $\mu_1, \mu_2, \ldots, \mu_n$. At certain points, we refer to the eigenvalues of $\boldsymbol{A}$ or $\boldsymbol{L}$ as the eigenvalues of G.

Degree

If G has maximum degree $\Delta(G)$, then $|\lambda| \leq \Delta(G)$ for every eigenvalue of G [83].

The sum of the Laplacian eigenvalues is equal to the degree sum of a graph [84], i.e., $\sum_{i=1}^{n} \mu_i = \sum_{i=1}^{n} d(i)$.

Regular Graphs

An r-regular graph G has row sums equal to r in the adjacency matrix of weights. The following results [76, 81, 83] also hold:

1. r is an eigenvalue of G;
2. For all eigenvalues λ of G, we have $|\lambda| \leq r$;
3. If r is an eigenvalue, then the all-1 vector is an eigenvector of G.

Connectedness

A graph G is *connected* if every pair of vertices of G is connected, i.e., there is a path between every two vertices of G. A $u-v$ *path* in a graph is a sequence of vertices beginning with u and ending at v such that consecutive vertices in the sequence are adjacent, with the additional restriction that no vertices are repeated [74].

If a graph G is connected, then: (1) the largest eigenvalue of $\boldsymbol{A}$ has multiplicity one, and (2) the second smallest eigenvalue of $\boldsymbol{L}$ is greater than zero [85, 86].

Closely related to idea of connectivity is the number of *components* of a graph. A component of G is a connected subgraph of G that is not a proper subgraph of any other connected subgraph of G [74]. The number of components of a graph G is denoted by $k(G)$. As related to spectra, $k(G)$ is equal to the multiplicity of the smallest eigenvalue $\mu = 0$ of the Laplacian $\boldsymbol{L}$ [86]. Thus, a graph is connected if and only if $k(G) = 1$, since it only has one component.

A graph G is *bipartite* if its vertex set can be partitioned into two distinct sets U and W such that every edge of G contains a vertex from U and a vertex from W [74]. As relating to spectra, a graph G is bipartite if and only if Spec $\boldsymbol{L}$ = Spec $\boldsymbol{L}_s$, where $\boldsymbol{L}_s$ is the signless Laplacian [81]. Recent research on internet topology has also revealed that a graph is bipartite if the normalized Laplacian has an eigenvalue of 2 [87, 88]. Additionally, an r-regular graph is bipartite if and only if $\lambda_1 = -r$ [89].

Diameter

Given a $u-v$ path, the *length* of a path is the number of edges between u and v. The *distance* between u and v is the length of the smallest $u-v$ path in a graph G. The *diameter* of a graph is the greatest distance between any two vertices of a connected G [74]. The diameter is often used to get a sense of how large a component is, especially useful when analyzing large networks. As relating to spectra [76, 77], if a connected graph G has d distinct eigenvalues, then its diameter is bounded above by $d-1$, i.e., $\text{diam}(G) \leq d-1$. This same result holds for Laplacian eigenvalues [81]. A lower bound on the diameter of a graph G of order n is also given in terms of the second smallest Laplacian eigenvalue [80], μ_2, as

$$\text{Diam}(G) \geq \frac{4}{n\mu_2}.$$

Second Smallest Eigenvalue

The second smallest eigenvalue of the Laplacian is an interesting topic in the field of spectral graph theory. For the remainder of this thesis, we refer to the second smallest eigenvalue of the Laplacian as μ_2, also called the *Fiedler value*. Miroslav Fiedler [90] referred to this eigenvalue as the *algebraic connectivity* of a graph G. As mentioned with regards to connectivity, a graph G is connected if and only if $\mu_2 > 0$. Another result [91] relates the algebraic connectivity with the number of vertices in a graph of degree $n-1$, i.e., $d^*_{n-1} \leq \mu_2$, where d^*_{n-1} is the number of vertices of degree $n-1$.

Fiedler also found relations between the algebraic connectivity and two graph parameters—vertex connectivity and edge connectivity. In order to understand these two parameters, we need the idea of cuts. A *vertex-cut* of G is a set U of vertices of G such that $G-U$ is disconnected, i.e., subtracting the set U (and the edges incident with these vertices) disconnects the graph G into components. Thus, the *vertex-connectivity* $\kappa(G)$ of a graph G is the cardinality of a minimum vertex-cut of G [74]. Fiedler [90] proved that if G is not a complete graph[7], then $\mu_2 \leq \kappa(G)$. Similarly, an *edge-cut* of G is a set X of edges of G such that $G-X$ is disconnected. Hence, the *edge-connectivity* $\eta(G)$ is the cardinality of a minimum edge-cut of G [74]. Once again, Fiedler [90] proved that $\mu_2 \leq \kappa(G) \leq \eta(G)$. We also have that $\mu_2 = n$ if and only if G is a complete graph on n vertices.

In graph theory and especially in network science, analysts and attackers are often concerned with cuts. In any model network, an adversary might want to know the minimum number of edges (links) or nodes to cut before the entire network is disconnected. This is a classic problem in graph theory, known as a type of *isoperimetric* problem. In spectral geometry, the isoperimetric problem is to find a closed curve of a given length that encloses the maximum area. In graph theory, this is equivalent to removing the smallest portion of a graph that disconnects it [79]. In 1970, Cheeger[8]derived bounds for μ_2 on a Riemannian bounded curve in terms of volumes and areas. Noga Alon and Vitali Milman [92] extended this to a graph, giving a bound for μ_2 in terms of edge cuts.

Consider a graph G with vertex set $V(G)$. We would like to split the graph into two dis-

[7] A complete graph of order n has $\binom{n}{2}$ edges and every two distinct vertices are adjacent.

[8] J. Cheeger wrote "A lower bound for the smallest eigenvalue of the Laplacian" in *Problems in Analysis*, 1970.

connected components via a cut, in this case an edge-cut. An edge-cut is defined as a bipartitition of $V(G)$, denoted by $E(S,\bar{S})$, where $S \subset V(G)$, $\bar{S} = V(G) \setminus S$, and $S \cap \bar{S} = \emptyset$. We also define the edge-cut $E(S,\bar{S})$ as the *edge boundary* ∂S of S. The cardinality of ∂S is the number of edges with one endpoint in S and another in $\bar{S}$. This quantity is then related with the *sizes* of S and $\bar{S}$, yielding a ratio of the proposed cut as

$$h_G(S) = \frac{|E(S,\bar{S})|}{min(|S|,|\bar{S}|)} = \frac{|\partial S|}{min(|S|,|\bar{S}|)}.$$

If we consider this formula for $h_G(S)$, then the Laplacian matrix is a better consideration. If using weights, it is often better to use the normalized Laplacian to account for the distribution of weights. In this alternate version denoted as $h'_G(S)$, the term *volume* is used instead to measure the size of S and $\bar{S}$. Let the volume of S be defined as $\text{vol}(S) = \sum_{v \in S} d(v)$. In an analogous manner,

$$h'_G(S) = \frac{|E(S,\bar{S})|}{min(\text{vol}(S),\text{vol}(\bar{S}))}.$$

As the term in the numerator decreases, the overall cut ratio decreases. Thus, an optimal edge-cut translates into removing the fewest edges. This minimum ratio is called the *Cheeger constant* of a graph, i.e.,

$$h_G = \inf_S h_G(S) \qquad \text{or} \qquad h'_G = \min_{\emptyset \subset S \subset V(G)} h'_G(S),$$

depending on which version of the Laplacian is used [79, 83]. Finding the minimum edge-cut is a nontrivial problem, especially when the order gets larger. From the Cheeger constant, we can formulate what is known as the *Cheeger inequality*.

Theorem 5.3.2. [79] Let $0 = \mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$ be the eigenvalues of the Laplacian and h_G be the Cheeger constant of a graph G. Then

$$2h_G \geq \mu_2 \geq \frac{h_G^2}{2\Delta(G)}, \tag{5.1}$$

where $\Delta(G)$ is the maximum degree of G. If using the normalized Laplacian, then the Cheeger inequality is given as

$$2h'_G \geq \mu_2 \geq \frac{h'^2_G}{2}. \tag{5.2}$$

This remarkable result gives us an upper bound for μ_2. In particular, when finding the Cheeger constant appears difficult, it can be estimated with μ_2. Control of μ_2 implies control of the Cheeger constant and hence edge-connectivity [93]. A small value for μ_2 implies a small number of edges needed to disconnect the graph; a large μ_2 implies many edges are required in an edge-cut. Cvetkovíc et al. [83] provided a similar result containing the edge boundary with the Laplacian eigenvalues:

$$\mu_2 \frac{|S||\bar{S}|}{n} \leq |\partial S| \leq \mu_n \frac{|S||\bar{S}|}{n} \implies \mu_2 \leq \frac{n|\partial S|}{|S||\bar{S}|} \leq \mu_n. \tag{5.3}$$

There are many other established bounds on the algebraic connectivity, but we only mention one more that relates the diameter and maximum degree of a graph. This result is due to Alon Nilli [94], although the notation is borrowed from [83]. If G is connected with maximum degree $\Delta(G)$ and diameter d, then

$$\mu_2 \leq \Delta(G) - 2\sqrt{\Delta(G)-1} + \frac{2\sqrt{\Delta(G)-1}-1}{\lfloor \frac{d}{2} \rfloor}.$$

Largest Eigenvalue

The largest eigenvalue of $\boldsymbol{A}$ is known as the *spectral radius* or *index* of G. Besides the other results already mentioned, for a connected graph G that is not regular, we have $d_{avg} \leq \lambda_n \leq \Delta(G)$, where d_{avg} this is the average degree of G, λ_n the spectral radius of G, and $\Delta(G)$ maximum degree of G, respectively [81].

Spanning Trees

A subgraph H of a graph G is a *spanning* subgraph if it spans all vertices in G, i.e., H and G have the same vertex set. If H is a tree[9], then it is called a *spanning tree*. Spanning trees have many applications in networks, from design to searching. The total number of spanning trees in a graph G, called the *complexity* of G, is determined by the Laplacian spectrum [83]. This result follows from the matrix-tree theorem.

Theorem 5.3.3. [80–83] Let G be a connected graph with Laplacian matrix $\boldsymbol{L}$ and eigen-

[9] A *tree* is a connected graph that does not contain cycles. A cycle is a closed circuit, in which vertices may be repeated but edges may not.

values $0 = \mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$. Then the number of spanning trees $\tau(G)$ of G is equal to any *cofactor* of $\boldsymbol{L}$. Symbolically,

$$\tau(G) = \det(\boldsymbol{L} + \frac{1}{n^2}J) = \frac{1}{n}\prod_{i=2}^{n}\mu_i, \tag{5.4}$$

where J is the all-ones matrix. The (i,j)-*cofactor* of a matrix M is given by $(-1)^{i+j}\det(M(i,j))$, and $M(i,j)$ is obtained by deleting row i and column j. It should also be noted that the following relationship also holds:

$$\text{adj}(\boldsymbol{L}) = \tau(G)\, J, \tag{5.5}$$

where $\text{adj}(\boldsymbol{L})$ is the *adjugate* matrix of $\boldsymbol{L}$, i.e., the transpose matrix of the cofactors. In this theorem, loops are ignored since a tree can not contain a closed path.

Cliques and Independence Number

A *clique* (pronounced `kleek` or `klik`) is a complete subgraph of a graph G [74]. This can also be thought of as a subset of the vertex set $V(G)$ in which all the vertices are pairwise adjacent. A *coclique* is a set of pairwise nonadjacent vertices in a graph G [81]. The *clique number* $\omega(G)$ a graph G is the order of the largest clique in G, while the *independence number* $\alpha(G)$ is the order of the largest coclique in G. We now present some bounds on these parameters with respect to eigenvalues of $\boldsymbol{A}$. Finding the clique number and independence number of a graph, along with many other graph invariants, are NP-complete[10]problems. However, determining the bounds on the eigenvalues can be performed in polynomial time.

Theorem 5.3.4. [83] Let G be a graph on n vertices. Let n^+ and n^- denote the number of positive and negative eigenvalues of the adjaceny matrix of G, respectively. Then

$$\alpha(G) \leq \min\{n - n^+,\, n - n^-\}. \tag{5.6}$$

[10]An NP-complete problem has a solution that can be verified in polynomial time, but there is no known algorithm that can find a solution in polynomial time. NP stands for *nondeterministic polynomial time*.

Theorem 5.3.5. [83] If G is regular, with ordinary spectrum $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, then

$$\alpha(G) \leq n\frac{-\lambda_1}{\lambda_n - \lambda_1}. \tag{5.7}$$

The clique number $\omega(G)$ is bounded above by the spectral radius of G [85], i.e., $\omega(G) \leq \lambda_n + 1$. Cvetkovíc et al. [83] provided a slight improvement on this bound.

Theorem 5.3.6. [83] Let m^-, m^0, m^+ denote the number of eigenvalues of a graph G which are less than, equal to, or greater than -1, respectively. Let $s = \min\{m^- + m^0 + 1,\ m^0 + m^+,\ 1 + \lambda_n\}$. Then $\omega(G) \leq s$. If $s = m^- + m^0 + 1$ and the eigenvalues greater than -1 exceed $m^- + m^0$, then $\omega(G) \leq s - 1$.

Theorem 5.3.7. [83, 95] If G is a graph with n vertices and m edges, then

$$\omega(G) \geq \frac{2m}{2m - \lambda_n^2}. \tag{5.8}$$

Chromatic Number

The *chromatic number* $\chi(G)$ of a graph is the smallest number of colors in a proper coloring of G. By a *proper* coloring, we mean an assignment of colors to the vertices of G, such that adjacent vertices are colored differently [74]. Determining the chromatic number of a graph is another decision problem, yet it is a classic exercise in graph theory.

Theorem 5.3.8. [81, 96] Let G be a connected graph with largest eigenvalue λ_n. Then $\chi(G) \leq \lambda_n + 1$, with equality if and only if G is complete or is an odd cycle.

Theorem 5.3.9. [81, 83] Let G be a graph with n vertices and at least one edge. Then

$$\chi(G) \geq 1 - \frac{\lambda_n}{\lambda_1} = 1 + \frac{\lambda_n}{|\lambda_1|}, \tag{5.9}$$

with equality if G is a nontrivial complete graph.

Vladimir Nikiforov [97] provided another lower bound on the chromatic number involving a Laplacian eigenvalue.

Theorem 5.3.10. [83, 97] Let G be a graph with n vertices. Then

$$\chi(G) \geq 1 + \frac{\lambda_n}{\mu_n - \lambda_1}. \tag{5.10}$$

Number of Walks

A $u-v$-*walk* in a graph G is a sequence of vertices beginning at u and ending at v such that consecutive vertices in the sequence are adjacent [74]. A k-walk is a walk of length k. Determining if a graph has a k-walk is an NP-complete problem as well.

Lemma 5.3.11. [76, 83] Let G be a graph with adjacency matrix $\boldsymbol{A}$. The number of walks of length k in G that start at vertex i and end at vertex j is given by the (i, j) entry $a_{ij}^{(k)}$ of the matrix $\boldsymbol{A}^k$.

A $u-v$-walk is *closed* if $u = v$. The number of closed walks of length k is given by [83]

$$\sum_{j=1}^{n} \lambda_j^k = \lambda_1^k + \lambda_2^k + \cdots + \lambda_n^k. \tag{5.11}$$

It follows from Lemma 5.3.11 that we can relate the eigenvalues to the number of triangles and edges in a graph. In particular, we have $\lambda_1^2 + \lambda_2^2 + \cdots + \lambda_n^2 = 2|E(G)|$, since the trace of $\boldsymbol{A}^2$ counts the number of closed walks of length two. Also, $\lambda_1^3 + \lambda_2^3 + \cdots + \lambda_n^3 = 6|T(G)|$, where $T(G)$ is the number of triangles in a graph.

In order to count the total number of walks of length k in a graph, we must first consider the product $\mathbf{j}^T \boldsymbol{A}^k \mathbf{j}$, where $\mathbf{j}$ is the all-ones vector of length n. Since $\boldsymbol{A}$ is a real symmetric matrix, its eigenvalues are associated with orthonormal eigenvectors. Thus, for choice of constants a_i, we can substitute for $\mathbf{j}$ with $\mathbf{j} = \mathbf{1} = \sum_i a_i \phi_i$, where ϕ_i is the eigenvector corresponding to λ_i. Utilizing this substitution [98], we have that the total number of walks of length k is

$$\left(\sum_i a_i \phi_i^T\right) A^k \left(\sum_i a_i \phi_i\right) = \left(\sum_i a_i \phi_i^T\right) \left(\sum_i a_i \lambda_i^k \phi_i\right) = \sum_i a_i^2 \lambda_i^k.$$

Alternate approaches to the total number of walks of length k are given in [77, 83].

Strongly Regular Graphs

A *strongly regular* graph is an r-regular graph on n vertices with the parameters (n,r,e,f) such that any two adjacent vertices have e common neighbors and any two nonadjacent vertices have f common neighbors [83]. Examples of strongly regular graphs include the 5-cycle C_5 with parameters $(5,2,0,1)$ and the Petersen graph with parameters $(10,3,0,1)$. The Petersen graph is referenced below in Figure 5.5.

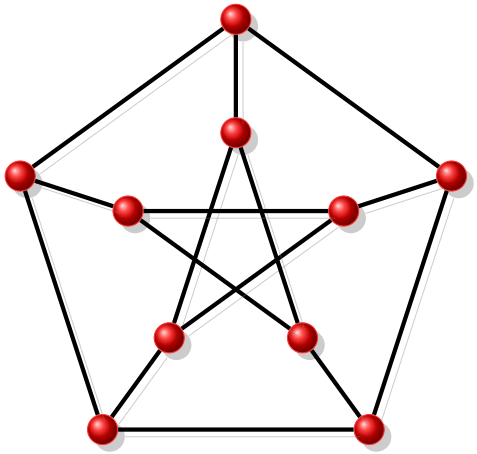


Figure 5.5: The Petersen Graph.

Theorem 5.3.12. [83,99] Let G be a connected r-regular graph, $r > 0$. Then G is strongly regular if and only if it has exactly three distinct eigenvalues. Furthermore, if these eigenvalues are $\lambda_1 = r$, $\lambda_2 = s$, and $\lambda_3 = t$, then

$$s,t = \frac{1}{2}(e-f) \pm \sqrt{\Delta} \qquad \Delta = (e-f)^2 + 4(r-f).$$

In the reverse direction, the parameters e and f are given in terms of the eigenvalues as

$$e = r+s+t+st, \quad f = r+st, \quad n = \frac{(r-s)(r-t)}{r+st}.$$

The multiplicities of r,s,t are $1,k,l$, respectively, where

$$k,l = \frac{1}{2}\left\{n-1 \mp \frac{2r+(n-1)(e-f)}{\sqrt{\Delta}}\right\}.$$

Furthermore, if $k = l$ (which only happens when Δ is not a perfect square), then the strongly regular graph is called a *conference graph*. If the graph is not a conference graph, then $\Delta = (s-t)^2$ is a perfect square, and r, s and t are all integers.

5.4 Cayley Graphs

Cayley graphs are named in honor of British mathematician Arthur Cayley (1821-1895). Among his many accomplishments, Cayley is best known for his work in developing modern group theory. Cayley is also credited for solidifying matrix theory and making discoveries in analytic geometry.

5.4.1 Definitions

We first need the idea of a *Cayley set* in order to define the Cayley graph that we need for a BF.

Definition 5.4.1. [39, 41] Let Γ be a group with identity element e. Suppose C is a subset of Γ. C is called a **Cayley set** if and only if whenever $g \in C$, then $g^{-1} \in C$, and $e \notin C$.

Definition 5.4.1 follows in the traditional manner of defining a generating set for a finite group, but we modify it by allowing the identity e to be an element of C. This exception allows for the presence of loops in the graph [41].

Definition 5.4.2. [41] The **Cayley graph** $G = G(\Gamma, C)$ of Γ with respect to C is the graph whose vertex set is Γ, with two vertices g and h adjacent if $gh^{-1} \in C$.

We now proceed to associate the Cayley graph to a BF, $f : \mathbb{F}_2^n \to \mathbb{F}_2$. Recall that $\mathbb{F}_2^n$ is a vector space, and for any vector $\boldsymbol{w} \in \mathbb{F}_2^n$, $\boldsymbol{w} = \boldsymbol{w}^{-1}$ with respect to the XOR operation. Since every vector is equal to its inverse in this group, any subset of $\mathbb{F}_2^n$ is a Cayley set. The subset we choose is the support of f, i.e., $\Omega_f = \{\boldsymbol{x} \in \mathbb{F}_2^n : f(\boldsymbol{x}) = 1\}$. We can now define a Cayley graph for a BF.

Definition 5.4.3. [39, 41] Let f be a BF on $\mathbb{F}_2^n$. Define the **Cayley graph** of f with respect to the set Ω_f as the graph $\Gamma_f = (\mathbb{F}_2^n, E_f)$. The vertex set of Γ_f is $\mathbb{F}_2^n$, while the edge set is

defined by

$$E_f = \{(\boldsymbol{w},\boldsymbol{u}) \in \mathbb{F}_2^n \times \mathbb{F}_2^n : \boldsymbol{w} \oplus \boldsymbol{u} \in \Omega_f\}$$
$$= \{(\boldsymbol{w},\boldsymbol{u}) \in \mathbb{F}_2^n \times \mathbb{F}_2^n : f(\boldsymbol{w} \oplus \boldsymbol{u}) = 1\}.$$

It follows from Definition 5.4.3 that the adjacency matrix $\boldsymbol{A}_f$ of Γ_f is the array of entries $a_{ij} = f(\boldsymbol{b}(i) \oplus \boldsymbol{b}(j))$, where $\boldsymbol{b}(i) = \alpha_i$ is the binary representation of the vector. The adjacency matrix $\boldsymbol{A}_f$ has the following properties [39, 41]:

i) The row sums of A_f are equal to $|\Omega_f|$;
ii) Property i) implies that Γ_f is a regular graph of degree $wt(f) = |\Omega_f|$;
iii) A_f has the *dyadic property* [100]: $a_{ij} = a_{i+2^{n-1},j+2^{n-1}}$, $0 \leq i, j \leq 2^{n-1}$;
iv) A_f is an $2^n \times 2^n$ symmetric matrix.

5.4.2 Boolean Cayley Graphs and their Spectra

For clarity, we now refer to Definition 5.4.3 as the one for Boolean Cayley graphs. BFs and their Walsh spectra have been analyzed extensively in the last 50 years, especially with regards to their associated cryptographic properties. The Cayley graph has also received much attention in the works of Laszlo Babai [101] and László Lovász [102], in particular with regards to its graph spectra. With the arrival of the Boolean Cayley graph, however, we now have a means to examine the graph spectra of a BF. The seminal work on Boolean Cayley graphs and their spectra was performed by Bernasconi and Codenotti [41], a summary of which is presented here.

Theorem 5.4.4. Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$, and let λ_i, $0 \leq i \leq 2^n - 1$, be the eigenvalues of the associated Cayley graph Γ_f. Then, there is a one-to-one correspondence between the spectrum of Γ_f and the Walsh spectrum of f, i.e., $\lambda_i = F(\boldsymbol{b}(i))$, for any i.

Proof: Recall that we defined the group character of $\mathbb{F}_2^n$ as the function $Q_{\boldsymbol{w}}(\boldsymbol{x}) = (-1)^{<\boldsymbol{w}\cdot\boldsymbol{x}>}$. The eigenvectors of Γ_f are equal to the characters $Q_{\boldsymbol{w}}(\boldsymbol{x})$. Then, the ith eigenvalue of $\boldsymbol{A}_f$,

corresponding to the eigenvector $Q_{\boldsymbol{b}(i)}$ is given by

$$\begin{aligned}\lambda_i &= \sum_x Q_{\boldsymbol{w}}(\boldsymbol{x}) f(\boldsymbol{x}) = \sum_x (-1)^{<\boldsymbol{w}\cdot\boldsymbol{x}>} f(\boldsymbol{x}) \\ &= \sum_x (-1)^{<\boldsymbol{b}(i)\cdot\boldsymbol{x}>} f(\boldsymbol{x}) = F(\boldsymbol{b}(i)). \qquad \blacksquare\end{aligned}$$

EXAMPLE 5.4.5. Let us use the function from Example 4.5.4, $f : \mathbb{F}_2^2 \to \mathbb{F}_2$ with ANF given by $1 \oplus x_1 \oplus x_2$.

$$F(\boldsymbol{w}) = W(f)(\boldsymbol{w}) = \sum_{\boldsymbol{x}\in\mathbb{F}_2^2} f(\boldsymbol{x}) \cdot (-1)^{<\boldsymbol{w}\cdot\boldsymbol{x}>}$$

$$\begin{aligned}F(00) &= 1(-1)^0 + 0 + 0 + 1(-1)^0 = 2 \\ F(01) &= 1(-1)^0 + 0 + 0 + 1(-1)^1 = 0 \\ F(10) &= 1(-1)^0 + 0 + 0 + 1(-1)^1 = 0 \\ F(11) &= 1(-1)^0 + 0 + 0 + 1(-1)^2 = 2\end{aligned}$$

$$\begin{aligned}\lambda_0 &= F(00) = 2 \\ \lambda_1 &= F(01) = 0 \\ \lambda_2 &= F(10) = 0 \\ \lambda_3 &= F(11) = 2\end{aligned}$$

We must be careful here not to confuse the subscript notation of the Cayley graph eigenvalues with the ordinary spectrum presented in Subsection 5.3.1. Translating the eigenvalues of this function to the spectrum notation of an adjacency matrix, we have

$$\text{Spec}\,\Gamma_f = \begin{pmatrix} 0 & 2 \\ 2 & 2 \end{pmatrix}.$$

Theorem 5.4.4 is a remarkable result not only because it links BFs to spectral graph theory, but it can save computational time. There are numerous computer programs that can

quickly compute the WT of a BF. In order to compute the eigenvalues of $\boldsymbol{A}_f$, however, we must first collect all of the vector combinations in the support of f and then create the $2^n \times 2^n$ matrix. For large n, this can be time consuming. For this thesis, in particular for Chapter 6, Theorem 5.4.4 only holds if we assign a weight of one to a loop in a pseudo-graph. If a loop is assigned a weight of two, then we do not see a one-to-one correspondence between the WT and the Cayley spectra.

Figure 5.6 depicts the Cayley graph from Example 5.4.5. Using some of the results from Section 5.3.1, we can make some comments about this graph. We know that the Cayley graph is regular, and using the adjacency matrix for this function, the row sums of $\boldsymbol{A}_f$ are two. Thus, Γ_f is 2-regular. Regularity also implies that $r = 2$ is an eigenvalue of Γ_f, and all other eigenvalues have absolute value less than or equal to 2. We can clearly see that the graph in Figure 5.6 is disconnected. This is verified because the largest eigenvalue $\lambda_3 = 2$ does not have multiplicity one. Also, the Laplacian eigenvalues (which in this case happen to be the same as the adjacency matrix) tell us that Γ_f is disconnected since the multiplicity of 0 implies that the graph has $k(G) = 2$ components. With regards to diameter, we do not define the diameter of a disconnected graph. However, the diameter of a component is possible to examine and since the components of the graph in Figure 5.6 are the same, we deduce that the diameter of a component is 1. This is verified with the eigenvalues of an adjacency matrix for one component, which are 0 and 2. The diameter is bounded above by $d - 1$, where $d = 2$ for the number of distinct eigenvalues. In this case, we know that $\text{diam}(G) \leq 2 - 1 = 1$. It is not very helpful to examine Γ_f with some of the other results since the graph is disconnected, but this will be looked at closer in Chapter 6.

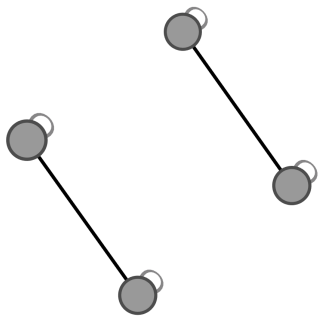


Figure 5.6: Cayley Graph Γ_f for the Function $1 \oplus x_1 \oplus x_2$.

Let $\langle \Omega_f \rangle \subseteq \mathbb{F}_2^n$ be the space of the $(0,1)$ sequences generated by Ω_f and let $\dim\langle \Omega_f \rangle$ be its dimension [39, 41]. Given this, observe that $\Omega_f = \{00, 11\}$ in Example 5.4.5. Since the zero vector is not part of a basis, this space has dimension one, i.e., $\dim\langle \Omega_f \rangle = 1$. With this new concept, we can state some more results on Boolean Cayley spectra, taken from [39,41].

5.4.3 Further Spectral Properties of Boolean Cayley Graphs

This section lists some other properties relating the Cayley spectra to graph properties as well as BF properties. For some of these results, it is assumed that $n \geq 4$, and these are marked with a $(*)$.

i*) The multiplicity of the largest spectral coefficient of f, $F(\boldsymbol{b}(0))$, is equal to $2^{n-\dim\langle \Omega_f \rangle}$.

ii) If $\dim\langle \Omega_f \rangle = n$, then Γ_f is connected.

iii*) If Γ_f is connected, then f has a spectral coefficient equal to -$wt(f)$ if and only if its Walsh spectrum is symmetric with respect to zero.

iv*) Γ_f is bipartite if and only if the Walsh spectrum of f is symmetric with respect to zero. Furthermore, Γ_f is bipartite if and only if $\mathbb{F}_2^n \backslash \Omega_f$ contains a subspace of dimension $n-1$.

v*) The number of nonzero spectral coefficients is equal to rank($\boldsymbol{A}_f$).

vi*) If Γ_f has two distinct eigenvalues, then its connected components are complete graphs and $\Omega_f \cup \{\boldsymbol{b}(0)\}$ is a group.

vii*) If Γ_f has three distinct eigenvalues none of which is zero, then these eigenvalues are

$$\lambda_0 = |\Omega_f| = wt(f), \quad \lambda_2 = -\lambda_1 = \sqrt{|\Omega_f| - e},$$

where e is the parameter of a strongly regular graph.

viii*) A BF defined on $\mathbb{F}_2^n$ (n even) is bent if and only if its associated Cayley graph Γ_f is a strongly regular graph with the additional property that $e = f$.

ix) Assume $n > 4$. If Γ_f is triangle free, then f is not bent.

x*) If Γ_f is the Cayley graph of f with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_v$ and g being the multiplicity of λ_1, then

$$\min\left\{g+1, 1-\frac{\lambda_1}{\lambda_{v-1}}\right\} \leq \chi(\Gamma_f) \leq |\Omega_f|,$$

provided $\lambda_{v-1} \neq 0$.

xi*) A BF is correlation immune of order ℓ if and only if the eigenvalues of its associated Cayley graph satisfy $\lambda_i = 0$ for all i with $1 \leq wt(\boldsymbol{b}(i)) \leq \ell$. Resiliency follows if $\lambda_0 = 2^{n-1}$.

CHAPTER 6:
DES Spectra

In this chapter, the S-Boxes of DES are examined in several ways. First, we find the BF representation for each of the coordinate functions within an S-Box. The relevant cryptographic properties of these functions are then computed and compared to each other. Second, we associate the BFs to a Cayley graph and examine the spectra of these graphs. With the spectra and cryptographic properties of the functions on hand, we can deduce some properties of the Cayley graph.

6.1 Methods

Recall from Chapter 4 that an S-Box is a function $f : \mathbb{F}_2^m \to \mathbb{F}_2^n$. For DES, this function is $F : \mathbb{F}_2^6 \to \mathbb{F}_2^4$. Each of the boxes contains four coordinate BFs, represented as $F(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), f_3(\boldsymbol{x}), f_4(\boldsymbol{x}))$, where each f_i is a mapping from the vector space $\mathbb{F}_2^6$ to the binary field $\mathbb{F}_2$, i.e., $f_i : \mathbb{F}_2^6 \to \mathbb{F}_2$. As an example of our approach, reconsider S-Box 1 from Table 3.10, displayed for the reader below.

S-Box 1								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	1110	0100	1101	0001	0010	1111	1011	1000
01	0000	1111	0111	0100	1110	0010	1101	0001
10	0100	0001	1110	1000	1101	0110	0010	1011
11	1111	1100	1000	0010	0100	0100	0001	0111
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	0011	1010	0110	1100	0101	1001	0000	0111
01	1010	0110	1100	1011	1001	0101	0011	1000
10	1111	1100	1001	0111	0011	1010	0101	0000
11	0101	1011	0011	1110	1010	0000	0110	1101

Since each coordinate function has a total of 2^6 input vectors, the S-Box entries represent

the 64 output bits to these functions. Thus, as a truth table function, f_1 has the following sequence of outputs: $(1,1,1,0,0,1,0,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,1,1,0,1,1,1,0,0,$ $0,0,0,1,1,1,0,1,0,0,1,1,0,1,1,0,0,0,1,0,1,1,0,0,1,0,0,0,0,0,1,1,1)$, corresponding to the entries of the first (00) row in S-Box 1.

The ordering of input variables we choose is in reverse order, i.e., $f(x_6,x_5,x_4,x_3,x_2,x_1)$. Again, the ordering of the variables is unimportant. Table 6.1 depicts the first 10 entries of the truth table for f_1 as an explanation of the variable ordering.

x_6	x_5	x_4	x_3	x_2	x_1	f
0	0	0	0	0	0	1
0	0	0	0	0	1	1
0	0	0	0	1	0	1
0	0	0	0	1	1	0
0	0	0	1	0	0	0
0	0	0	1	0	1	1
0	0	0	1	1	0	0
0	0	0	1	1	1	0
0	0	1	0	0	0	1
0	0	1	0	0	1	1

Table 6.1: First 10 Truth Table Entries for S-Box 1.

The unique truth table output is then input into a software program to compute the various cryptographic properties of the BFs. For this thesis, multiple programs are used for analysis in order to verify accuracy, and these include SageMathCloud™, R© and R-Studio©, as well as Boolean Functions Workshop 1.3©. The adjacency matrix A_f is then formed from the definitions in Chapter 5. Note that for any vector $\boldsymbol{w}$ in $\mathbb{F}_2^6$, $\boldsymbol{w}\oplus\boldsymbol{w}=\mathbf{0}$ over the binary field $\mathbb{F}_2$. Thus, since an edge $(\boldsymbol{w},\boldsymbol{u})$ is present in the associated Cayley graph if $f(\boldsymbol{w}\oplus\boldsymbol{u})=1$, then $f(\boldsymbol{w}\oplus\boldsymbol{w})=1$ implies the presence of a loop. Hence, if the first output in a function's truth table sequence is a one, then the associated Cayley graph has a loop at every vertex.

The adjacency matrix is then input into MATLAB®, where the eigenvalues are computed

and compared to the corresponding function's WT for verification. The adjacency matrix is also imported into MATLAB® and Gephi© to produce a graph.

6.2 DES S-Box Spectra

This section details the results obtained via the methods in Section 6.1. Each S-Box is given its own subsection for reader clarity. Recall that the notation we adopt for spectra is given by Definition 5.3.1.

6.2.1 S-Box 1

The ANFs for the coordinate functions are displayed in Table 6.2.

Function	ANF	Number of Terms	Degree
f_1	$1 \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_4 \oplus x_2x_5 \oplus x_4x_5 \oplus x_2x_6 \oplus x_5x_6 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_3x_4x_5 \oplus x_1x_2x_6 \oplus x_2x_3x_6 \oplus x_1x_4x_6 \oplus x_2x_4x_6 \oplus x_1x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6$	31	5
f_2	$x_3 + x_5 + x_6 + x_1x_4 + x_2x_4 + x_3x_4 + x_1x_6 + x_5x_6 + x_1x_2x_4 + x_2x_3x_4 + x_1x_2x_5 + x_2x_3x_5 + x_1x_4x_5 + x_3x_4x_5 + x_2x_3x_6 + x_2x_5x_6 + x_4x_5x_6 + x_1x_2x_4x_5 + x_2x_3x_4x_5 + x_1x_2x_4x_6 + x_1x_3x_4x_6 + x_2x_3x_4x_6 + x_1x_2x_5x_6 + x_2x_3x_5x_6 + x_1x_4x_5x_6 + x_2x_4x_5x_6 + x_1x_2x_3x_4x_6 + x_1x_2x_4x_5x_6$	28	5
f_3	$x_1 + x_4 + x_5 + x_6 + x_1x_2 + x_1x_3 + x_1x_4 + x_1x_5 + x_2x_5 + x_3x_5 + x_1x_6 + x_4x_6 + x_2x_3x_4 + x_1x_4x_5 + x_1x_2x_6 + x_1x_3x_6 + x_2x_3x_6 + x_2x_4x_6 + x_3x_4x_6 + x_1x_5x_6 + x_1x_2x_3x_5 + x_1x_3x_5x_6 + x_1x_4x_5x_6 + x_1x_2x_3x_4x_5 + x_1x_2x_3x_4x_6 + x_1x_2x_3x_5x_6$	26	5
f_4	$1 + x_5 + x_6 + x_2x_3 + x_1x_4 + x_2x_4 + x_3x_4 + x_1x_5 + x_3x_5 + x_1x_6 + x_3x_6 + x_1x_2x_4 + x_1x_3x_4 + x_2x_3x_4 + x_1x_2x_5 + x_2x_4x_5 + x_2x_3x_6 + x_3x_4x_6 + x_1x_5x_6 + x_3x_5x_6 + x_4x_5x_6 + x_1x_2x_3x_5 + x_1x_2x_4x_5 + x_2x_3x_4x_5 + x_1x_2x_3x_6 + x_1x_2x_4x_6 + x_1x_3x_4x_6 + x_1x_2x_5x_6 + x_1x_3x_5x_6 + x_1x_4x_5x_6 + x_2x_4x_5x_6 + x_1x_2x_3x_4x_5 + x_1x_2x_4x_5x_6$	33	5

Table 6.2: ANF and Degree of S-Box 1 BFs.

Tables 6.3, 6.4, 6.5, and 6.6 display the various spectra for these same functions as well as their relevant cryptographic criteria.

Function	Walsh Spectra and Walsh-Hadamard Spectra
f_1	W: $(32,0,0,0,0,0,-4,-4,2,2,-2,-2,-2,-2,-2,-2,0,0,4,-4,8,0,0,0,-2,6,-2,-2,2,2,-2,6,2,2,2,2,$ $6,6,2,2,0,0,4,-12,-8,8,0,0-2,-10,2,2,2,2,10,2,0,0,8,0,8,0,-4,-4)$ WH: $(0,0,0,0,0,0,8,8,-4,-4,4,4,4,4,4,4,0,0,-8,8,-16,0,0,0,4,-12,4,4,-4,-4,4,-12,-4,-4,-4,-4,$ $-12,-12,-4,-4,0,0,-8,24,16,-16,0,0,4,20,-4,-4,-4,-4,-20,-4,0,0,-16,0,-16,0,8,8)$
f_2	W: $(32,0,0,0,2,2,2,2,0,4,0,-4,-6,6,2,6,2,2,2,-6,-8,0,0,0,-2,2,-2,2,-4,0,-4,0,0,-4,0,$ $-4,2,-2,2,-2,0,8,0,0,-6,-6,2,-6,-2,-6,-2,2,-4,0,-12,0,2,-6,2,10,-8,0,8,0)$ WH: $(0,0,0,0,-4,-4,-4,-4,0,-8,0,8,12,-12,-4,-12,-4,-4,-4,12,16,0,0,0,4,-4,4,-4,8,0,8,0,0,8,0,$ $8,-4,4,-4,4,0,-16,0,0,12,12,-4,12,4,12,4,-4,8,0,24,0,-4,12,-4,-20,16,0,-16,0)$
f_3	W: $(32,0,0,0,4,-4,0,0,2,-2,2,-2,2,-2,-2,2,2,-2,6,2,2,6,2,-2,-4,-4,0,0,0,0,0,-8,-2,2,2,-2,$ $-2,2,6,-6,-4,-12,0,0,0,0,8,8,-8,0,0,0,4,-4,0,-8,-2,2,-10,2,-10,2,2,-2)$ WH: $(0,0,0,0,-8,8,0,0,-4,4,-4,4,-4,4,4,-4,-4,4,-12,-4,-4,-12,-4,4,8,8,0,0,0,0,0,16,4,-4,-4,4,$ $4,-4,-12,12,8,24,0,0,0,0,-16,-16,16,0,0,0,-8,8,0,16,4,-4,20,-4,20,-4,-4,4)$
f_4	W: $(32,0,0,0,-2,-2,-2,-2,0,0,4,4,6,-2,2,-6,4,4,0,0,2,-6,-2,6,8,-8,0,0,-2,-2,-2,-2,-2,-2,2,$ $2,0,0,4,4,6,-2,6,-2,0,0,-8,-8,-2,6,6,-2,8,8,0,0,2,2,-2,-2,4,4,-8,8)$ WH: $(0,0,0,0,4,4,4,4,0,0,-8,-8,-12,4,-4,12,-8,-8,0,0,-4,12,4,-12,-16,16,0,0,4,4,4,4,4,4,-4,$ $-4,0,0,-8,-8,-12,4,-12,4,0,0,16,16,4,-12,-12,4,-16,-16,0,0,-4,-4,4,4,-8,-8,16,-16)$

Table 6.3: Walsh Spectra and Walsh-Hadamard Spectra of S-Box 1 BFs.

Function	Cayley Graph Spectra ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	Distinct λ_i
f_1	$\begin{pmatrix} -12 & -10 & -8 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 1 & 1 & 5 & 11 & 18 & 15 & 2 & 4 & 4 & 1 & 1 \end{pmatrix}$	12
f_2	$\begin{pmatrix} -12 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 2 & 7 & 6 & 6 & 19 & 16 & 1 & 2 & 2 & 1 & 1 \end{pmatrix}$	12
f_3	$\begin{pmatrix} -12 & -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 1 & 2 & 3 & 1 & 5 & 11 & 18 & 15 & 2 & 3 & 2 & 1 \end{pmatrix}$	12
f_4	$\begin{pmatrix} -8 & -6 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 4 & 2 & 18 & 15 & 6 & 8 & 6 & 4 & 1 \end{pmatrix}$	9

Table 6.4: Cayley Graph Spectra of S-Box 1 BFs.

Function	**Laplacian Spectra** ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 40 & 42 & 44 \\ 1 & 1 & 4 & 4 & 2 & 15 & 18 & 11 & 5 & 1 & 1 & 1 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 44 \\ 1 & 1 & 2 & 2 & 1 & 16 & 19 & 6 & 6 & 7 & 2 & 1 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 & 44 \\ 1 & 2 & 3 & 2 & 15 & 18 & 11 & 5 & 1 & 3 & 2 & 1 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 38 & 40 \\ 1 & 4 & 6 & 8 & 6 & 15 & 18 & 2 & 4 \end{pmatrix}$

Table 6.5: Laplacian Spectra of Cayley Graphs Associated with S-Box 1 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	5	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	20	20	20	24
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.6: Cryptographic Properties of S-Box 1 BFs.

Figure 6.1 represents the Cayley graph for the first row BF. Due to software limitations, loops are not present.

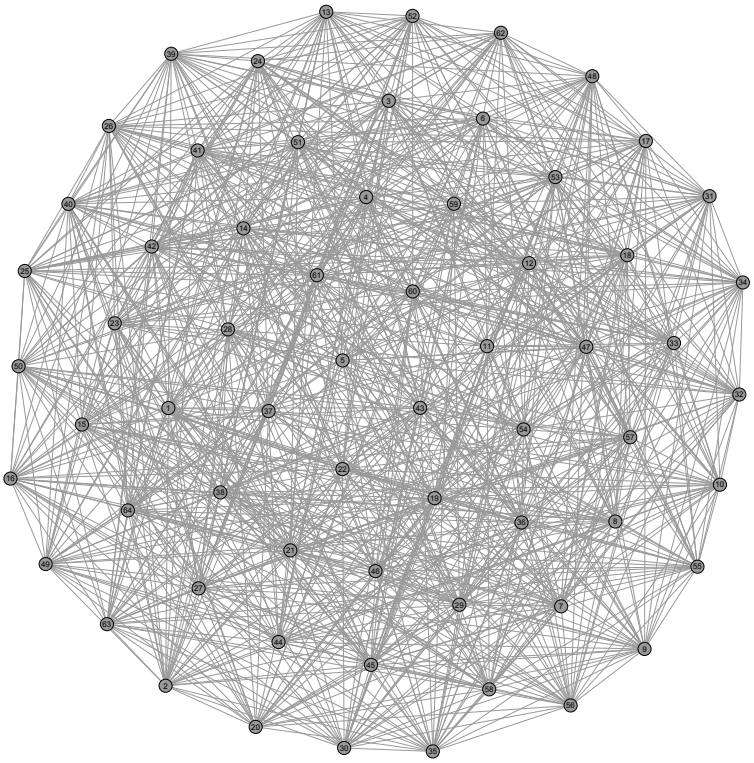


Figure 6.1: Cayley Graph Representation for f_1 of S-Box 1, Loops Not Present.

Spectral Observations

Here we state some observations from the Cayley graphs of S-Box 1 with the relations given in Chapter 5.

Regularity: The Cayley graphs associated with all of the 32 BFs are regular of degree $wt(f) = |\Omega_f| = 32$.

Connectivity: This is apparent from the first graph in Figure 6.1, but all graphs Γ_f are

connected since the multiplicity of $\lambda_n = 32$ is 1 (also $\mu_2 > 0$ and $\dim\langle\Omega_f\rangle = 6$). Additionally, since $m(\mu_1) = 1$, then Γ_f has one component. Since none of the Walsh spectra are symmetric with respect to zero, we do not see any Cayley spectra where $\lambda_i = -32$.

Bipartite: None of the graphs are bipartite since the Walsh spectra is not symmetric with respect to 0.

Rank: The ranks of the adjacency matrices $\boldsymbol{A}_{f_i}$ are equal to 46, 45, 46, and 49, respectively.

Diameter: The diameters of the Cayley graphs associated with S-Box 1 are bounded according to the following inequalities:

$$\begin{aligned}
f_1: &\quad \frac{4}{64\cdot 22} = 0.0028 \leq \mathrm{Diam}(\Gamma_f) \leq 12-1 = 11;\\
f_2: &\quad \frac{4}{64\cdot 22} = 0.0028 \leq \mathrm{Diam}(\Gamma_f) \leq 12-1 = 11;\\
f_3: &\quad \frac{4}{64\cdot 24} = 0.0026 \leq \mathrm{Diam}(\Gamma_f) \leq 12-1 = 11;\\
f_4: &\quad \frac{4}{64\cdot 24} = 0.0026 \leq \mathrm{Diam}(\Gamma_f) \leq 9-1 = 8.
\end{aligned}$$

Using SageMathCloud™, we determine the diameter to be 2 for all four of the Cayley graphs.[11]

Edge Connectivity: Since μ_2 is 22 or 24, we have an idea for the number of edges needed in an edge-cut of the Cayley graphs.

Spanning Trees: The Cayley graphs for these functions have large complexities. Chapter 5 provided a formula for the number of spanning trees in a graph in terms of the nonzero Laplacian eigenvalues. It is also known that for r-regular graphs of order n, the complexity of the graph G is bounded above [76] by

$$\tau(G) \leq \frac{1}{n}\left(\frac{nr}{n-1}\right)^{n-1}.$$

The number of spanning trees in these graphs are approximately $2.277\times 10^{92}, 1.731\times 10^{93}, 1.7648\times 10^{93}$, and 2.2708×10^{92}. These complexities achieve close to the upper bound of 2.8129×10^{93}, but interestingly Γ_{f_4} has the smallest complexity and

[11] For many of these graph properties, we consider only the underlying simple graph for those pseudographs with loops, since many graph parameters are only defined on simple graphs.

also the smallest number of distinct eigenvalues (and consequently a tighter upper bound on diameter).

Clique and Independence Number: We have bounds for the clique number based off the details in Chapter 5, and these are universal for all of the S-Boxes since they are in terms of the spectral radius. Thus, we have $2 \leq \omega(\Gamma_f) \leq 33$ for the entire set of S-Boxes. This bound is not ideal, since we would like a tighter interval. Methods are available, however, for computing the clique number of a graph with the aid of NetworkX© and Python™. Using SageMathCloud™, we compute the clique number to be 8 for all four graphs, i.e., $\omega(\Gamma_f) = 8$. For the independence number, we have an upper bound based on the inequality in Chapter 5 for regular graphs. Hence, $\alpha(\Gamma_f)$ is bounded above by 17.4545, 17.4545, 17.4545, and 12.8, respectively. Using the Independent Set Algorithm© by Dharwadker [103], however, the independence number is found to be $\alpha(\Gamma_f) = 8$ for the S-Box 1 Cayley graphs.

Chromatic Number: The bounds for $\chi(G)$ given in Chapter 5 give us that $3.\overline{6} \leq \chi(\Gamma_f) \leq 32$. We can increase the lower bound slightly since it is known that $\frac{n}{\alpha} \leq \chi$. Hence, $8 \leq \chi(\Gamma_f) \leq 32$. Using SageMathCloud™, we compute the chromatic number also to be 8 for all four graphs.

6.2.2 S-Box 2

In this subsection, we mimic the approach taken in Subsection 6.2.1, with less explanation. S-Box 2 is displayed in Table 6.7.

S-Box 2								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	1111	0001	1000	1110	0110	1011	0011	0100
01	0011	1101	0100	0111	1111	0010	1000	1110
10	0000	1110	0111	1011	1010	0100	1101	0001
11	1101	1000	1010	0001	0011	1111	0100	0010
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	1001	0111	0010	1101	1100	0000	0101	1010
01	1100	0000	0001	1010	0110	1001	1011	0101
10	0101	1000	1100	0110	1001	0011	0010	1111
11	1011	0110	0111	1100	0000	0101	1110	1001

Table 6.7: S-Box 2 in Binary Form.

The BFs in S-Box 2 are converted to their ANFs in Table 6.8. Tables 6.9, 6.10, 6.11, and 6.12 follow in the same manner as before.

Function	ANF	Number of Terms	Degree
f_1	$1 \oplus x_3 \oplus x_5 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_4x_6 \oplus x_5x_6 \oplus x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_2x_3x_5 \oplus x_2x_4x_5 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_2x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6$	28	5
f_2	$x_2 \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_2x_5 \oplus x_4x_6 \oplus x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_2x_3x_4 \oplus x_2x_3x_5 \oplus x_2x_3x_6 \oplus x_1x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_3x_4x_6$	23	4
f_3	$x_3 \oplus x_5 \oplus x_1x_4 \oplus x_2x_4 \oplus x_1x_5 \oplus x_1x_6 \oplus x_4x_6 \oplus x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_2x_3x_4 \oplus x_2x_3x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	21	5
f_4	$1 \oplus x_2 \oplus x_5 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_3x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_3x_6 \oplus x_4x_6 \oplus x_1x_2x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_3x_6 \oplus x_2x_3x_6 \oplus x_1x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_3x_4x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_2x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	33	5

Table 6.8: ANF and Degree of S-Box 2 BFs.

Function	Walsh Spectra and Walsh-Hadamard Spectra
f_1	W: $(32,0,0,0,0,0,0,0,2,-2,-2,2,6,2,10,-2,4,0,0,4,-4,8,0,4,2,-6,-6,2,6,-2,-2,6,$ $2,2,-2,-2,2,2,-2,-2,4,0,-4,0,0,-4,0,-12,-2,2,6,2,6,-6,6,2,-4,-4,0,0,8,8,-4,-4)$ WH: $(0,0,0,0,0,0,0,0,-4,4,4,-4,-12,-4,-20,4,-8,0,0,-8,8,-16,0,-8,-4,12,12,-4,-12,4,4,-12,$ $-4,-4,4,4,-4,-4,4,4,-8,0,8,0,0,8,0,24,4,-4,-12,-4,-12,12,-12,-4,8,8,0,0,-16,-16,8,8)$
f_2	W: $(32,0,0,0,0,0,0,0,0,0,0,0,8,0,0,-8,-4,-4,0,0,-4,-4,0,0,0,0,4,4,0,8,-4,4,4,0,0,-4,-4,0,0,4,$ $4,0,-8,4,4,0,-8,4,4,-8,-4,0,-4,8,-4,-8,0,4,0,4,-8,-4,-8,-4)$ WH: $(0,0,0,0,0,0,0,0,0,0,0,0,0,-16,0,0,16,8,8,0,0,8,8,0,0,0,0,-8,-8,0,-16,8,-8,-8,0,0,8,8,0,0,-8,$ $-8,0,16,-8,-8,0,16,-8,-8,16,8,0,8,-16,8,16,0,-8,0,-8,16,8,16,8)$
f_3	W: $(32,0,0,0,-2,-2,2,2,-6,2,2,2,0,0,-4,4,0,0,4,-4,-2,-10,-2,-2,-2,-2,2,2,-4,-4,-4,-4,0,0,0,0,$ $2,2,-2,-2,-2,6,6,-10,-8,8,-4,4,4,4,-8,0,-10,-2,-2,-2,-2,-2,2,2,0,0,8,8)$ WH: $(0,0,0,0,4,4,-4,-4,12,-4,-4,-4,0,0,8,-8,0,0,-8,8,4,20,4,4,4,4,-4,-4,8,8,8,8,0,0,0,0,$ $-4,-4,4,4,4,-12,-12,20,16,-16,8,-8,-8,-8,16,0,20,4,4,4,4,4,-4,-4,0,0,-16,-16)$
f_4	W: $(32,0,0,0,2,2,-2,-2,2,2,-2,-2,2,-4,0,-4,8,2,2,2,2,8,0,-4,4,0,4,4,0,6,-6,6,$ $2,-2,2,-2,2,0,-4,4,0,8,0,4,4,2,-6,-6,-6,-4,0,4,8,2,6,6,-6,-6,-6,6,-2,0,-8,0,0)$ WH: $(0,0,0,0,-4,-4,4,4,-4,4,4,-4,8,0,8,-16,-4,-4,-4,-4,-16,0,8,-8,0,-8,-8,0,-12,12,-12,$ $-4,4,-4,4,-4,0,8,-8,0,-16,0,-8,-8,-4,12,12,12,8,0,-8,-16,-4,-12,-12,12,12,12,-12,4,0,16,0,0)$

Table 6.9: Walsh Spectra and Walsh-Hadamard Spectra of S-Box 2 BFs.

Function	Cayley Graph Spectra ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	Distinct λ_i
f_1	$\begin{pmatrix} -12 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 3 & 7 & 10 & 16 & 12 & 4 & 6 & 3 & 1 & 1 \end{pmatrix}$	11
f_2	$\begin{pmatrix} -8 & -4 & 0 & 4 & 8 & 32 \\ 7 & 12 & 29 & 12 & 3 & 1 \end{pmatrix}$	6
f_3	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 3 & 2 & 1 & 7 & 15 & 14 & 11 & 5 & 2 & 3 & 1 \end{pmatrix}$	11
f_4	$\begin{pmatrix} -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 1 & 7 & 5 & 7 & 14 & 13 & 7 & 5 & 4 & 1 \end{pmatrix}$	10

Table 6.10: Cayley Graph Spectra of S-Box 2 BFs.

Function	Laplacian Spectra ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 44 \\ 1 & 1 & 3 & 6 & 4 & 12 & 16 & 10 & 7 & 3 & 1 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 24 & 28 & 32 & 36 & 40 \\ 1 & 3 & 12 & 29 & 12 & 7 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 2 & 3 & 5 & 11 & 14 & 15 & 7 & 1 & 2 & 3 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 \\ 1 & 4 & 5 & 7 & 13 & 14 & 7 & 5 & 7 & 1 \end{pmatrix}$

Table 6.11: Laplacian Spectra of Cayley Graphs Associated with S-Box 2 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	4	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	20	24	22	24
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.12: Cryptographic Properties of S-Box 2 BFs.

Figure 6.2 represents the Cayley graph for the second row BF. Since all of these Cayley graphs are 32-regular, we omit the remaining graphical representations from this thesis.

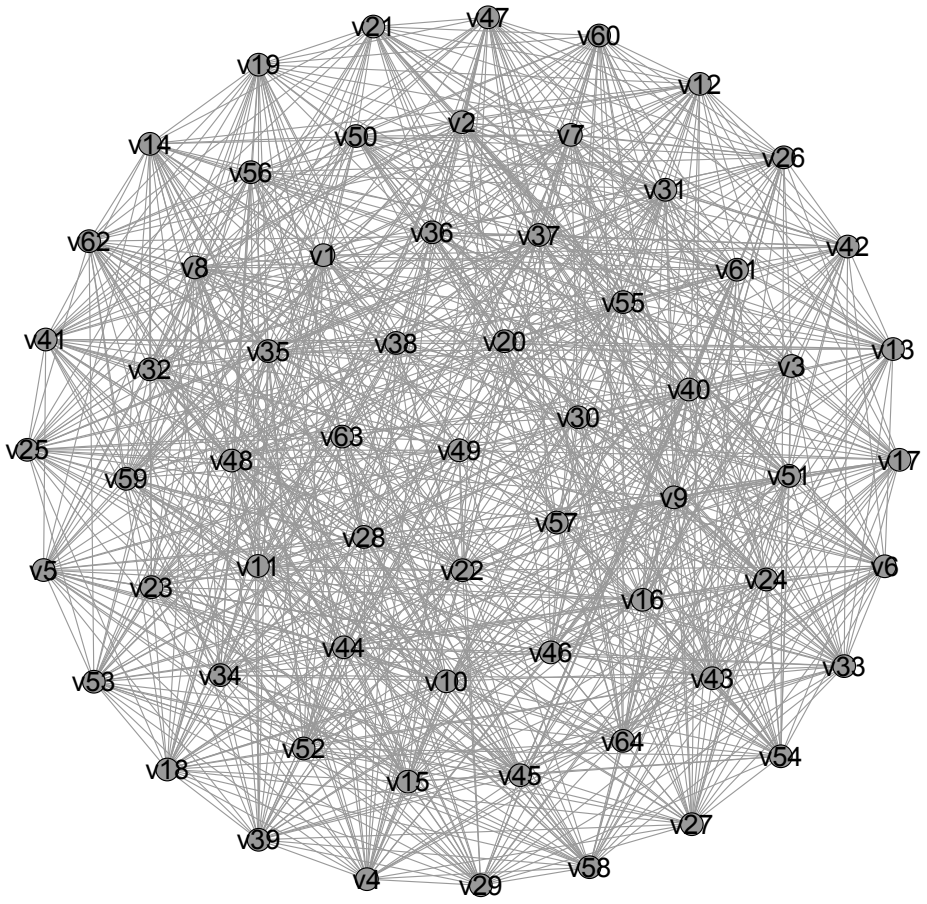


Figure 6.2: Cayley Graph Representation for f_2 of S-Box 1.

Spectral Observations

We deviate here for the second S-Box and present the results in table format without bounds where appropriate.

Graph Parameter	Γ_{f_1}	Γ_{f_2}	Γ_{f_3}	Γ_{f_4}
Regularity; deg	Yes; 32	Yes; 32	Yes; 32	Yes; 32
Connected; $k(\Gamma_f)$	Yes; 1	Yes; 1	Yes; 1	Yes; 1
Bipartite	No	No	No	No
Rank(A_{f_i}**)**	48	35	50	50
Diameter	2	2	2	2
Spanning Trees; $\tau(\Gamma_f)$	2.2642×10^{92}	1.7368×10^{93}	1.8851×10^{93}	2.2737×10^{92}
Clique Number	8	8	8	8
Independence Number	8	8	8	8
Chromatic Number	8	8	8	8

Table 6.13: Properties of Cayley Graphs Associated with S-Box 2 BFs.

6.2.3 S-Box 3

S-Box 3 is displayed in Table 6.14.

S-Box 3								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	1010	0000	1001	1110	0110	0011	1111	0101
01	1101	0111	0000	1001	0011	0100	0110	1010
10	1101	0110	0100	1001	1000	1111	0011	0000
11	0001	1010	1101	0000	0110	1001	1000	0111
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	0001	1101	1100	0111	1011	0100	0010	1000
01	0010	1000	0101	1110	1100	1011	1111	0001
10	1011	0001	0010	1100	0101	1010	1110	0111
11	0100	1111	1110	0011	1011	0101	0010	1100

Table 6.14: S-Box 3 in Binary Form.

The BFs in S-Box 3 are converted to their ANFs in Table 6.15. Tables 6.16, 6.17, 6.18, and 6.19 follow in the same manner as before.

Function	ANF	Number of Terms	Degree
f_1	$1 \oplus x_1 \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_1x_3 \oplus x_2x_4 \oplus x_3x_4 \oplus x_2x_5 \oplus x_3x_5 \oplus x_4x_5 \oplus x_1x_6 \oplus x_4x_6 \oplus x_2x_3x_4 \oplus x_1x_4x_5 \oplus x_1x_2x_6 \oplus x_1x_3x_6 \oplus x_2x_3x_6 \oplus x_3x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_3x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_2x_3x_4x_5 \oplus x_1x_2x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_4x_5x_6$	30	5
f_2	$1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_4 \oplus x_3x_5 \oplus x_4x_5 \oplus x_4x_6 \oplus x_1x_2x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_2x_4x_5 \oplus x_3x_4x_5 \oplus x_1x_4x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_2x_3x_4x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6$	29	5
f_3	$1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_1x_5 \oplus x_3x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_5x_6 \oplus x_1x_2x_3 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_2x_3x_5 \oplus x_2x_4x_5 \oplus x_1x_5x_6 \oplus x_3x_5x_6 \oplus x_1x_2x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_2x_3x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6$	29	5
f_4	$x_3 \oplus x_4 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_1x_6 \oplus x_5x_6 \oplus x_1x_2x_3 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_2x_4x_5 \oplus x_1x_4x_6 \oplus x_2x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_2x_5x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	26	5

Table 6.15: ANF and Degree of S-Box 3 BFs.

Function	Walsh Spectra and Walsh-Hadamard Spectra
f_1	W: $(32,0,0,0,2,6,-2,2,-4,0,-4,0,2,2,-2,-2,0,0,4,4,-6,-2,2,6,-4,0,0,4,2,2,-6,-6,$ $2,2,-2,-2,4,0,4,0,-6,6,-2,-6,0,0,8,-8,-6,10,2,2,4,0,-8,4,2,-2,2,-2,8,8,4,4)$ WH: $(0,0,0,0,-4,-12,4,-4,8,0,8,0,-4,-4,4,4,0,0,-8,-8,12,4,-4,-12,8,0,0,-8,-4,-4,12,12,$ $-4,-4,4,4,-8,0,-8,0,12,-12,4,12,0,0,-16,16,12,-20,-4,-4,-8,0,16,-8,-4,4,-4,4,-16,-16,-8,-8)$
f_2	W: $(32,0,0,0,0,-4,0,-4,0,0,0,0,0,4,0,4,-2,-2,2,2,-6,-2,-2,2,2,2,-2,-2,6,2,2,-2,-2,-2,-2,$ $-2,-2,2,-2,2,6,-10,-2,-2,6,2,-2,10,4,-4,0,8,0,4,12,0,8,0,4,-4,-4,-8,8,4)$ WH: $(0,0,0,0,0,8,0,8,0,0,0,0,0,-8,0,-8,4,4,-4,-4,12,4,4,-4,-4,-4,4,4,-12,-4,-4,4,4,4,4,$ $4,4,-4,4,-4,-12,20,4,4,-12,-4,4,-20,-8,8,0,-16,0,-8,-24,0,-16,0,-8,8,8,16,-16,-8)$
f_3	W: $(32,0,0,0,0,0,0,0,4,0,0,4,0,-4,4,8,-2,-2,2,2,2,2,-2,-2,2,-2,-6,-2,10,6,2,6,$ $-2,-2,2,2,-2,-2,2,2,6,2,6,-6,-6,6,2,6,4,-4,4,-4,0,-8,8,0,-4,0,0,-4,4,-8,-8,4)$ WH: $(0,0,0,0,0,0,0,0,-8,0,0,-8,0,8,-8,-16,4,4,-4,-4,-4,-4,4,4,-4,4,12,4,-20,-12,-4,-12,$ $4,4,-4,-4,4,4,-4,-4,-12,-4,-12,12,12,-12,-4,-12,-8,8,-8,8,0,16,-16,0,8,0,0,8,-8,16,16,-8)$
f_4	W: $(32,0,0,0,-2,2,2,-2,2,-2,-2,2,-4,-4,-4,-4,0,0,0,0,2,-10,6,2,-2,2,2,-2,-12,-4,-4,4,-4,0,0,$ $4,2,-6,2,2,-2,6,-2,-2,0,-4,-4,-8,-4,0,0,4,6,-2,-10,6,2,2,-6,2,0,4,4,8)$ WH: $(0,0,0,0,4,-4,-4,4,-4,4,4,-4,8,8,8,8,0,0,0,0,-4,20,-12,-4,4,-4,-4,4,24,8,8,-8,8,0,0,$ $-8,-4,12,-4,-4,4,-12,4,4,0,8,8,16,8,0,0,-8,-12,4,20,-12,-4,-4,12,-4,0,-8,-8,-16)$

Table 6.16: Walsh Spectra and Walsh-Hadamard Spectra of S-Box 3 BFs.

Function	Cayley Graph Spectra ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	Distinct λ_i
f_1	$\begin{pmatrix} -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 2 & 6 & 3 & 9 & 14 & 13 & 9 & 3 & 3 & 1 & 1 \end{pmatrix}$	11
f_2	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 12 & 32 \\ 1 & 1 & 1 & 5 & 16 & 15 & 10 & 6 & 3 & 3 & 1 & 1 & 1 \end{pmatrix}$	13
f_3	$\begin{pmatrix} -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 3 & 3 & 5 & 10 & 14 & 12 & 7 & 6 & 2 & 1 & 1 \end{pmatrix}$	11
f_4	$\begin{pmatrix} -12 & -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 1 & 2 & 1 & 2 & 10 & 10 & 13 & 14 & 5 & 4 & 1 & 1 \end{pmatrix}$	12

Table 6.17: Cayley Graph Spectra of S-Box 3 BFs.

Function	Laplacian Spectra ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 \\ 1 & 1 & 3 & 3 & 9 & 13 & 14 & 9 & 3 & 6 & 2 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 20 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 1 & 1 & 3 & 3 & 6 & 10 & 15 & 16 & 5 & 1 & 1 & 1 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 \\ 1 & 1 & 2 & 6 & 7 & 12 & 14 & 10 & 5 & 3 & 3 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 & 44 \\ 1 & 1 & 4 & 5 & 14 & 13 & 10 & 10 & 2 & 1 & 2 & 1 \end{pmatrix}$

Table 6.18: Laplacian Spectra of Cayley Graphs Associated with S-Box 3 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	5	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	22	20	22	20
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.19: Cryptographic Properties of S-Box 3 BFs.

Spectral Observations

Table 6.20 depicts the relevant properties of the Cayley graphs associated with the S-Box 3 BFs.

Graph Parameter	Γ_{f_1}	Γ_{f_2}	Γ_{f_3}	Γ_{f_4}
Regularity; deg	Yes; 32	Yes; 32	Yes; 32	Yes; 32
Connected; $k(\Gamma_f)$	Yes; 1	Yes; 1	Yes; 1	Yes; 1
Bipartite	No	No	No	No
Rank(A_{f_i})	50	49	50	51
Diameter	2	2	2	2
Spanning Trees; $\tau(\Gamma_f)$	2.2695×10^{92}	2.2106×10^{92}	2.2699×10^{92}	1.761×10^{93}
Clique Number	8	8	8	8
Independence Number	8	8	8	8
Chromatic Number	8	8	8	8

Table 6.20: Properties of Cayley Graphs Associated with S-Box 3 BFs.

6.2.4 S-Box 4

S-Box 4 is displayed in Table 6.21.

S-Box 4								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	0111	1101	1110	0011	0000	0110	1001	1010
01	1101	1000	1011	0101	0110	1111	0000	0011
10	1010	0110	1001	0000	1100	1011	0111	1101
11	0011	1111	0000	0110	1010	0001	1101	1000
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	0001	0010	1000	0101	1011	1100	0100	1111
01	0100	0111	0010	1100	0001	1010	1110	1001
10	1111	0001	0011	1110	0101	0010	1000	0100
11	1001	0100	0101	1011	1100	0111	0010	1110

Table 6.21: S-Box 4 in Binary Form.

Table 6.22 lists the ANFs for the BFs of S-Box 4.

Function	ANF	Number of Terms	Degree
f_1	$x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_3x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_3x_6 \oplus x_5x_6 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_2x_3x_5 \oplus x_1x_3x_6 \oplus x_2x_3x_6 \oplus x_3x_4x_6 \oplus x_2x_5x_6 \oplus x_3x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_2x_3x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6$	35	5
f_2	$1 \oplus x_2 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_3x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_5x_6 \oplus x_1x_3x_4 \oplus x_1x_2x_5 \oplus x_2x_3x_5 \oplus x_1x_3x_6 \oplus x_2x_3x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_1x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6$	29	5
f_3	$1 \oplus x_1 \oplus x_3 \oplus x_2x_3 \oplus x_2x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_3x_5 \oplus x_4x_5 \oplus x_1x_6 \oplus x_4x_6 \oplus x_5x_6 \oplus x_1x_3x_4 \oplus x_1x_3x_5 \oplus x_1x_4x_5 \oplus x_2x_4x_5 \oplus x_3x_4x_5 \oplus x_2x_3x_6 \oplus x_2x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6$	32	5
f_4	$x_2 \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_2x_3 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_1x_6 \oplus x_4x_6 \oplus x_5x_6 \oplus x_1x_3x_4 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_3x_4x_6 \oplus x_2x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6$	28	5

Table 6.22: ANF and Degree of S-Box 4 BFs.

Figure 6.3 displays the Walsh-Hadamard spectra of the S-Box 4 BFs obtained from R©. It is assumed that the reader can easily compute the Walsh spectra via the relation in Equation 4.16.

```
> wh(s4r1)
 [1]   0   0   0   0   8   0   0  -8   4   4   4   4  -4   4  20  -4   0   8   8
[20]   0  -8  -8  -8   8  -4   4   4  -4   4  20   4   4  -4  -4   4   4  -4   4
[39]  -4   4   0  16  -8   8  16  -8   0   8 -20   4  -4   4  -4  -4   4  20  -8
[58]   0  -8 -16  -8  -8  16   0
> wh(s4r2)
 [1]   0   0   0   0   8   0   0   8  -4   4  -4   4   4  20  -4  -4   0   8  -8
[20]   0  -8  -8   8  -8   4   4  -4  -4  -4   4 -20   4  -4   4   4  -4  -4  -4
[39]  -4  -4  -8  -8 -16   0  -8   0   8  16  -4  -4  -4 -20 -20   4   4  -4  16
[58]  -8   0  -8   0  16   8  -8
> wh(s4r3)
 [1]   0   0   0   0  -8   0   0  -8  -4  -4   4   4  -4   4 -20   4   0  -8   8
[20]   0  -8  -8   8  -8  -4   4  -4   4  -4 -20   4   4  -4  -4  -4  -4   4  -4
[39]  -4   4   0 -16  -8   8  16  -8   0  -8  20  -4  -4   4  -4  -4  -4 -20  -8
[58]   0   8  16   8   8  16   0
> wh(s4r4)
 [1]   0   0   0   0   8   0   0  -8  -4   4   4  -4  -4 -20  -4  -4   0   8   8
[20]   0   8   8   8  -8  -4  -4  -4  -4  -4   4  20  -4   4  -4   4  -4  -4  -4
[39]   4   4  -8  -8  16   0   8   0   8  16  -4  -4   4  20  20  -4   4  -4 -16
[58]   8   0  -8   0  16  -8   8
```

Figure 6.3: Walsh-Hadamard Spectra of S-Box 4 BFs.

Tables 6.23, 6.24, and 6.25 follow in the same manner as before.

Function	**Cayley Graph Spectra** ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	**Distinct** λ_i
f_1	$\begin{pmatrix} -10 & -8 & -4 & -2 & 0 & 2 & 4 & 8 & 10 & 32 \\ 3 & 3 & 6 & 17 & 11 & 11 & 10 & 1 & 1 & 1 \end{pmatrix}$	10
f_2	$\begin{pmatrix} -10 & -8 & -4 & -2 & 0 & 2 & 4 & 8 & 10 & 32 \\ 1 & 3 & 6 & 11 & 11 & 17 & 10 & 1 & 3 & 1 \end{pmatrix}$	10
f_3	$\begin{pmatrix} -10 & -8 & -4 & -2 & 0 & 2 & 4 & 8 & 10 & 32 \\ 1 & 3 & 6 & 11 & 11 & 17 & 10 & 1 & 3 & 1 \end{pmatrix}$	10
f_4	$\begin{pmatrix} -10 & -8 & -4 & -2 & 0 & 2 & 4 & 8 & 10 & 32 \\ 3 & 3 & 10 & 9 & 11 & 19 & 6 & 1 & 1 & 1 \end{pmatrix}$	10

Table 6.23: Cayley Graph Spectra of S-Box 4 BFs.

Function	Laplacian Spectra ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 22 & 24 & 28 & 30 & 32 & 34 & 36 & 40 & 42 \\ 1 & 1 & 1 & 10 & 11 & 11 & 17 & 6 & 3 & 3 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 22 & 24 & 28 & 30 & 32 & 34 & 36 & 40 & 42 \\ 1 & 3 & 1 & 10 & 17 & 11 & 11 & 6 & 3 & 1 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 22 & 24 & 28 & 30 & 32 & 34 & 36 & 40 & 42 \\ 1 & 3 & 1 & 10 & 17 & 11 & 11 & 6 & 3 & 1 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 22 & 24 & 28 & 30 & 32 & 34 & 36 & 40 & 42 \\ 1 & 1 & 1 & 6 & 19 & 11 & 9 & 10 & 3 & 3 \end{pmatrix}$

Table 6.24: Laplacian Spectra of Cayley Graphs Associated with S-Box 4 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	5	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	22	22	22	22
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.25: Cryptographic Properties of S-Box 4 BFs.

Spectral Observations

Table 6.26 depicts the relevant properties of the Cayley graphs associated with the S-Box 4 BFs.

Graph Parameter	Γ_{f_1}	Γ_{f_2}	Γ_{f_3}	Γ_{f_4}
Regularity; deg	Yes; 32	Yes; 32	Yes; 32	Yes; 32
Connected; $k(\Gamma_f)$	Yes; 1	Yes; 1	Yes; 1	Yes; 1
Bipartite	No	No	No	No
Rank(A_{f_i}**)**	53	53	53	53
Diameter	2	2	2	2
Spanning Trees; $\tau(\Gamma_f)$	1.7454×10^{93}	2.26×10^{92}	2.26×10^{92}	1.7523×10^{93}
Clique Number	8	8	8	8
Independence Number	8	8	8	8
Chromatic Number	8	8	8	8

Table 6.26: Properties of Cayley Graphs Associated with S-Box 4 BFs.

6.2.5 S-Box 5

S-Box 5 is displayed in Table 6.27.

S-Box 5								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	0010	1100	0100	0001	0111	1010	1011	0110
01	1110	1011	0010	1100	0100	0111	1101	0001
10	0100	0010	0001	1011	1010	1101	0111	1000
11	1011	1000	1100	0111	0001	1110	0010	1101
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	1000	0101	0011	1111	1101	0000	1110	1001
01	0101	0000	1111	1010	0011	1001	1000	0110
10	1111	1001	1100	0101	0110	0011	0000	1110
11	0110	1111	0000	1001	1010	0100	0101	0011

Table 6.27: S-Box 5 in Binary Form.

Table 6.28 lists the ANFs for the BFs of S-Box 5.

Function	ANF	Number of Terms	Degree
f_1	$x_2 \oplus x_3 \oplus x_6 \oplus x_1x_2 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_4x_5 \oplus x_1x_6 \oplus x_4x_6 \oplus x_1x_2x_3 \oplus x_1x_3x_4 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_3x_4x_5 \oplus x_2x_3x_6 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_3x_4x_5 \oplus x_1x_2x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_4x_5x_6$	27	5
f_2	$1 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_1x_6 \oplus x_5x_6 \oplus x_1x_3x_4 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_1x_2x_6 \oplus x_1x_4x_6 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_2x_5x_6 \oplus x_3x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_3x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_4x_5x_6$	30	5
f_3	$x_1 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_4 \oplus x_3x_4 \oplus x_4x_5 \oplus x_1x_6 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_4x_5 \oplus x_2x_4x_5 \oplus x_1x_2x_6 \oplus x_1x_4x_6 \oplus x_2x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	30	5
f_4	$1 \oplus x_1 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_3x_5 \oplus x_2x_6 \oplus x_3x_6 \oplus x_1x_2x_4 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_3x_4x_5 \oplus x_1x_2x_6 \oplus x_1x_3x_6 \oplus x_3x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_3x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_2x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	39	5

Table 6.28: ANF and Degree of S-Box 5 BFs.

Figure 6.4 displays the Walsh-Hadamard spectra of the S-Box 5 BFs obtained from R©. It is again assumed that the reader can compute the Walsh spectra via the relation in Equation 4.16.

```
> wh(s5r1)
 [1]   0   0   0   0  -4  -4   4   4   8   0  -8   0  -4   4   4  -4   8   8   0   0  20 -12   4   4   0
[26]  -8  -8   0   4 -20   4  -4   4  -4   4  12   0   8   8   0 -12 -12   4   4   8   8  16  16  12   4
[51] -12  -4  -8   0  -8  16  -4  -4   4   4   0   0  16 -16
> wh(s5r2)
 [1]   0   0   0   0  -4   4  -4   4   0   8   8   0   4   4  -4  12  -4 -12  -4   4  -8   8   8   8   4
[26]   4  -4  -4  -8   0 -16  -8  -4   4  -4   4   8  -8  -8   8 -12 -12  -4  12   8   0 -16  -8   0 -16
[51]   0   0  12 -12  12   4 -16  -8   8 -16 -12   4  -4  12
> wh(s5r3)
 [1]   0   0   0   0   4   4  -4  12  -4  -4   4   4  -8  -8  -8   8   0   8   0  -8  -4   4 -12  -4   4
[26]  -4  -4   4  -8  16   8   0   4  -4   4  -4   0   8   8   0   8   0 -16   8  12 -12  -4 -12  12  -4
[51]  12  12  16  16  -8  -8   8   8   0  16 -12   4 -12   4
> wh(s5r4)
 [1]   0   0   0   0  12  -4   4   4  -4 -12  -4   4  -8   0  16  -8  -4  -4  -4  -4   0   0  -8   8  -8
[26]   0   8   0  -4   4   4  12  -4  -4  -4  -4   0   0   8  -8   8   0   8 -16  -4  20   4  -4   0   0
[51]   0   0 -20  -4 -12 -12  12 -12  -4 -12  -8 -16  16   8
```

Figure 6.4: Walsh-Hadamard Spectra of S-Box 5 BFs.

Tables 6.29, 6.30, and 6.31 follow in the same manner as before.

Function	**Cayley Graph Spectra** ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	**Distinct** λ_i
f_1	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 4 & 2 & 7 & 15 & 14 & 9 & 5 & 4 & 1 & 1 & 1 \end{pmatrix}$	12
f_2	$\begin{pmatrix} -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 5 & 9 & 11 & 10 & 11 & 7 & 5 & 5 & 1 \end{pmatrix}$	9
f_3	$\begin{pmatrix} -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 4 & 5 & 9 & 11 & 10 & 11 & 7 & 5 & 1 & 1 \end{pmatrix}$	10
f_4	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 2 & 3 & 6 & 6 & 15 & 16 & 6 & 5 & 2 & 1 & 1 \end{pmatrix}$	12

Table 6.29: Cayley Graph Spectra of S-Box 5 BFs.

Function	Laplacian Spectra ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 1 & 1 & 4 & 5 & 9 & 14 & 15 & 7 & 2 & 4 & 1 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 \\ 1 & 5 & 5 & 7 & 11 & 10 & 11 & 9 & 5 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 \\ 1 & 1 & 5 & 7 & 11 & 10 & 11 & 9 & 5 & 4 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 1 & 2 & 5 & 6 & 16 & 15 & 6 & 6 & 3 & 2 & 1 \end{pmatrix}$

Table 6.30: Laplacian Spectra of Cayley Graphs Associated with S-Box 5 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	5	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	22	24	24	22
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.31: Cryptographic Properties of S-Box 5 BFs.

Spectral Observations

Table 6.32 depicts the relevant properties of the Cayley graphs associated with the S-Box 5 BFs.

Graph Parameter	Γ_{f_1}	Γ_{f_2}	Γ_{f_3}	Γ_{f_4}
Regularity; deg	Yes; 32	Yes; 32	Yes; 32	Yes; 32
Connected; $k(\Gamma_f)$	Yes; 1	Yes; 1	Yes; 1	Yes; 1
Bipartite	No	No	No	No
Rank(A_{f_i})	50	54	54	49
Diameter	2	2	2	2
Spanning Trees; $\tau(\Gamma_f)$	1.7206×10^{93}	2.2469×10^{92}	1.7337×10^{93}	2.286×10^{92}
Clique Number	8	8	8	8
Independence Number	8	8	8	8
Chromatic Number	8	8	8	8

Table 6.32: Properties of Cayley Graphs Associated with S-Box 5 BFs.

6.2.6 S-Box 6

S-Box 6 is displayed in Table 6.33.

S-Box 6								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	1100	0001	1010	1111	1001	0010	0110	1000
01	1010	1111	0100	0010	0111	1100	1001	0101
10	1001	1110	1111	0101	0010	1000	1100	0011
11	0100	0011	0010	1100	1001	0101	1111	1010
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	0000	1101	0011	0100	1110	0111	0101	1011
01	0110	0001	1101	1110	0000	1011	0011	1000
10	0111	0000	0100	1010	0001	1101	1011	0110
11	1011	1110	0001	0111	0110	0000	1000	1101

Table 6.33: S-Box 6 in Binary Form.

Table 6.34 lists the ANFs for the BFs of S-Box 6.

Function	ANF	Number of Terms	Degree
f_1	$1 \oplus x_2 \oplus x_3 \oplus x_6 \oplus x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_4x_5 \oplus x_2x_6 \oplus x_5x_6 \oplus x_1x_2x_3 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_2x_4x_5 \oplus x_3x_4x_5 \oplus x_1x_4x_6 \oplus x_1x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_2x_3x_4x_5 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	31	5
f_2	$1 \oplus x_1 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_1x_3 \oplus x_2x_5 \oplus x_3x_5 \oplus x_2x_6 \oplus x_5x_6 \oplus x_1x_2x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_2x_3x_6 \oplus x_1x_4x_6 \oplus x_1x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_3x_4x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	30	5
f_3	$1 \oplus x_1 \oplus x_2 \oplus x_5 \oplus x_6 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_3x_5 \oplus x_4x_5 \oplus x_5x_6 \oplus x_1x_2x_3 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_2x_4x_5 \oplus x_3x_4x_5 \oplus x_1x_2x_6 \oplus x_1x_4x_6 \oplus x_2x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_4x_5x_6$	36	5
f_4	$x_1 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_2x_5 \oplus x_3x_5 \oplus x_4x_6 \oplus x_1x_2x_3 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_2x_3x_6 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_3x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_3x_4x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	29	5

Table 6.34: ANF and Degree of S-Box 6 BFs.

Figure 6.5 displays the Walsh-Hadamard spectra of the S-Box 6 BFs obtained from R©. It is again assumed that the reader can compute the Walsh spectra via the relation in Equation 4.16.

```
> wh(s6r1)
 [1]   0   0   0   0   4  -4  -4   4   4   4  -4  -4   0  -8 -16  -8   4   4  -4  -4   8 -16   8   0   8   8  -8  -8
[29]   4  12  -4  20   4 -12  -4  -4  -8   0  -8   0   8   8   8  -8 -12  12  -4 -12 -16   0   0   0   4  -4 -20   4
[57]   4   4  -4  12 -16  -8   0   8
> wh(s6r2)
 [1]   0   0   0   0   4   4   4   4  -4  -4   4   4   8  -8   0  16  -4  -4  -4  12   0   0 -16   0   0   0   8  -8
[29]  -4 -20   4   4  -4   4  -4   4   0  -8   0  -8 -16  -8  -8   0  -4   4  20  -4   8 -16   8   0  12   4  -4   4
[57] -12  -4  -4 -12  16  -8  -8 -16
> wh(s6r3)
 [1]   0   0   0   0  -4   4   4  -4   8   0   0  -8   4   4  20   4  -4   4  -4   4  -8   8   0   0  -4  -4   4   4
[29]  -8   0  -8 -16  -4  -4  -4  -4   0  -8  -8   0   4 -20  -4   4   8   8   8  -8 -16   8   0  -8   4 -12  12 -20
[57]   0   0  -8  -8  20  12 -12  -4
> wh(s6r4)
 [1]   0   0   0   0   4  -4   4  -4   4   4   4   4  -8   0  -8   0   0   0   8   8  12   4   4  -4 -12 -12  -4  -4
[29]   0   8  -8   0   0   0   0   0  -4   4  -4   4   4  20   4 -12  16  -8 -16  -8  16   0  -8   8   4  -4  -4  20
[57]   4   4  12  12  -8  16 -16   8
```

Figure 6.5: Walsh-Hadamard Spectra of S-Box 6 BFs.

Tables 6.35, 6.36, and 6.37 follow in the same manner as before.

Function	**Cayley Graph Spectra** ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	**Distinct** λ_i
f_1	$\begin{pmatrix} -10 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 3 & 8 & 12 & 11 & 12 & 8 & 3 & 4 & 1 & 1 \end{pmatrix}$	11
f_2	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 2 & 2 & 4 & 13 & 13 & 13 & 8 & 2 & 4 & 1 & 1 \end{pmatrix}$	12
f_3	$\begin{pmatrix} -10 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 2 & 2 & 6 & 12 & 13 & 12 & 10 & 2 & 2 & 2 & 1 \end{pmatrix}$	11
f_4	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 2 & 3 & 3 & 5 & 15 & 14 & 9 & 7 & 3 & 2 & 1 \end{pmatrix}$	11

Table 6.35: Cayley Graph Spectra of S-Box 6 BFs.

Function	Laplacian Spectra ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 42 \\ 1 & 1 & 4 & 3 & 8 & 12 & 11 & 12 & 8 & 3 & 1 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 1 & 4 & 2 & 8 & 13 & 13 & 13 & 4 & 2 & 2 & 1 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 42 \\ 1 & 2 & 2 & 2 & 10 & 12 & 13 & 12 & 6 & 2 & 2 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 2 & 3 & 7 & 9 & 14 & 15 & 5 & 3 & 3 & 2 \end{pmatrix}$

Table 6.36: Laplacian Spectra of Cayley Graphs Associated with S-Box 6 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	5	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	22	22	22	22
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.37: Cryptographic Properties of S-Box 6 BFs.

Spectral Observations

Table 6.38 depicts the relevant properties of the Cayley graphs associated with the S-Box 6 BFs.

Graph Parameter	Γ_{f_1}	Γ_{f_2}	Γ_{f_3}	Γ_{f_4}
Regularity; deg	Yes; 32	Yes; 32	Yes; 32	Yes; 32
Connected; $k(\Gamma_f)$	Yes; 1	Yes; 1	Yes; 1	Yes; 1
Bipartite	No	No	No	No
Rank(A_{f_i}**)**	53	51	51	50
Diameter	2	2	2	2
Spanning Trees; $\tau(\Gamma_f)$	2.2498×10^{92}	2.2657×10^{92}	2.2628×10^{92}	1.7426×10^{93}
Clique Number	8	8	8	8
Independence Number	8	8	8	8
Chromatic Number	8	8	8	8

Table 6.38: Properties of Cayley Graphs Associated with S-Box 6 BFs.

6.2.7 S-Box 7

S-Box 7 is displayed in Table 6.39.

S-Box 7								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	0100	1011	0010	1110	1111	0000	1000	1101
01	1101	0000	1011	0111	0100	1001	0001	1010
10	0001	0100	1011	1101	1100	0011	0111	1110
11	0110	1011	1101	1000	0001	0100	1010	0111
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	0011	1100	1001	0111	0101	1010	0110	0001
01	1110	0011	0101	1100	0010	1111	1000	0110
10	1010	1111	0110	1000	0000	0101	1001	0010
11	1001	0101	0000	1111	1110	0010	0011	1100

Table 6.39: S-Box 7 in Binary Form.

Table 6.40 lists the ANFs for the BFs of S-Box 7.

Function	ANF	Number of Terms	Degree
f_1	$x_1 \oplus x_3 \oplus x_5 \oplus x_1x_2 \oplus x_1x_4 \oplus x_2x_4 \oplus x_1x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_4x_6 \oplus x_5x_6 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_3x_4x_5 \oplus x_1x_2x_6 \oplus x_2x_4x_6 \oplus x_2x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_4x_5x_6$	27	5
f_2	$1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_1x_2 \oplus x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_2x_6 \oplus x_4x_6 \oplus x_1x_2x_3 \oplus x_2x_4x_5 \oplus x_2x_4x_6 \oplus x_1x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_1x_2x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_4x_5x_6$	24	5
f_3	$x_4 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_2x_5 \oplus x_3x_5 \oplus x_1x_6 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_1x_2x_6 \oplus x_1x_4x_6 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_3x_5x_6 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_1x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_4x_5x_6$	28	5
f_4	$x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_2x_3 \oplus x_1x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_3x_5 \oplus x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_3x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_2x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_1x_2x_4x_5x_6$	32	5

Table 6.40: ANF and Degree of S-Box 7 BFs.

Figure 6.6 displays the Walsh-Hadamard spectra of the S-Box 7 BFs obtained from R©. It is again assumed that the reader can compute the Walsh spectra via the relation in Equation 4.16.

```
> wh(s7r1)
 [1]   0   0   0   0   4   4   4   4   0   0   0   0 -12  20   4   4  -4  -4   4   4  16   0   8  -8   4   4  -4  -4   8  -8
[31]   0  16   0  -8  -8   0   4  -4  -4   4   0   8   8   0 -12  -4 -20   4   4  -4   4  12   8  16  -8  16  -4   4  -4 -12
[61]  16   8 -16  -8
> wh(s7r2)
 [1]   0   0   0   0   4   4  -4  -4   0   0   0   0  -4  -4 -12  20  -8   8   0   0 -20  -4  -4  -4   8  -8   0   0 -12   4
[31] -12 -12   4   4   4 -12  -8   8   0   0  12  12 -20  -4  -8   8   0   0  -4  -4   4   4   0 -16   0 -16   4   4  -4  -4
[61]   0  16  16   0
> wh(s7r3)
 [1]   0   0   0   0   4  -4   4  -4   8   8   0   0  12 -12 -12  -4  -4  -4  -4  -4   0  -8  16   8  -4  -4   4   4   0   8
[31]   8 -16  -8   8   0   0  -4   4   4  -4  16   0   0   0 -12  12 -12  -4  12  12   4 -12   0  -8   8 -16  12  12  -4  12
[61]   0   8  16   8
> wh(s7r4)
 [1]   0   0   0   0   4  -4   4  -4   4   4   4   4  -8   0  -8   0  -4   4  -4 -12   8   8  -8   8  -8   0   8   0   4 -12
[31]   4   4   0   0   0   0  -4   4  -4   4   4   4   4   4  16   8  16   8  -4 -12  -4   4 -16  16   0  16  -8 -16   8  16
[61]  12  -4 -20  12
```

Figure 6.6: Walsh-Hadamard Spectra of S-Box 7 BFs.

Tables 6.41, 6.42, and 6.43 follow in the same manner as before.

Function	**Cayley Graph Spectra** ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	**Distinct** λ_i
f_1	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 5 & 1 & 6 & 16 & 13 & 10 & 6 & 3 & 1 & 1 & 1 \end{pmatrix}$	12
f_2	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 2 & 2 & 4 & 10 & 19 & 12 & 4 & 5 & 2 & 2 & 1 \end{pmatrix}$	12
f_3	$\begin{pmatrix} -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 3 & 7 & 9 & 7 & 14 & 13 & 3 & 5 & 2 & 1 \end{pmatrix}$	10
f_4	$\begin{pmatrix} -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 5 & 2 & 7 & 17 & 12 & 9 & 5 & 3 & 2 & 1 & 1 \end{pmatrix}$	11

Table 6.41: Cayley Graph Spectra of S-Box 7 BFs.

Function	**Laplacian Spectra** ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 1 & 1 & 3 & 6 & 10 & 13 & 16 & 6 & 1 & 5 & 1 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 2 & 2 & 5 & 4 & 12 & 19 & 10 & 4 & 2 & 2 & 1 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 \\ 1 & 2 & 5 & 3 & 13 & 14 & 7 & 9 & 7 & 3 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 \\ 1 & 1 & 2 & 3 & 5 & 9 & 12 & 17 & 7 & 2 & 5 \end{pmatrix}$

Table 6.42: Laplacian Spectra of Cayley Graphs Associated with S-Box 7 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	5	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	22	22	24	22
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.43: Cryptographic Properties of S-Box 7 BFs.

Spectral Observations

Table 6.44 depicts the relevant properties of the Cayley graphs associated with the S-Box 7 BFs.

Graph Parameter	Γ_{f_1}	Γ_{f_2}	Γ_{f_3}	Γ_{f_4}
Regularity; deg	Yes; 32	Yes; 32	Yes; 32	Yes; 32
Connected; $k(\Gamma_f)$	Yes; 1	Yes; 1	Yes; 1	Yes; 1
Bipartite	No	No	No	No
Rank(A_{f_i}**)**	51	45	50	52
Diameter	2	2	2	2
Spanning Trees; $\tau(\Gamma_f)$	1.727×10^{93}	2.2533×10^{92}	1.7258×10^{93}	1.7076×10^{93}
Clique Number	8	8	8	8
Independence Number	8	8	8	8
Chromatic Number	8	8	8	8

Table 6.44: Properties of Cayley Graphs Associated with S-Box 7 BFs.

6.2.8 S-Box 8

S-Box 8 is displayed in Table 6.45.

S-Box 8								
ROW/COL	0000	0001	0010	0011	0100	0101	0110	0111
00	1101	0010	1000	0100	0110	1111	1011	0001
01	0001	1111	1101	1000	1010	0011	0111	0100
10	0111	1011	0100	0001	1001	1100	1110	0010
11	0010	0001	1110	0111	0100	1010	1000	1101
ROW/COL	1000	1001	1010	1011	1100	1101	1110	1111
00	1010	1001	0011	1110	0101	0000	1100	0111
01	1100	0101	0110	1011	0000	1110	1001	0010
10	0000	0110	1010	1101	1111	0011	0101	1000
11	1111	1100	1001	0000	0011	0101	0110	1011

Table 6.45: S-Box 8 in Binary Form.

Table 6.46 lists the ANFs for the BFs of S-Box 8.

Function	ANF	Number of Terms	Degree
f_1	$1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_1x_2 \oplus x_1x_4 \oplus x_1x_5 \oplus x_4x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_3x_6 \oplus x_4x_6 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_2x_4x_5 \oplus x_1x_2x_6 \oplus x_2x_3x_6 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_2x_4x_5 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_4x_5x_6$	31	5
f_2	$x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_1x_2 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_2x_6 \oplus x_1x_2x_3 \oplus x_1x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_2x_6 \oplus x_1x_3x_6 \oplus x_2x_3x_6 \oplus x_1x_4x_6 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_5x_6 \oplus x_1x_2x_3x_4 \oplus x_1x_2x_3x_5 \oplus x_1x_2x_4x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_2x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_4x_5x_6$	30	5
f_3	$x_1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_1x_2 \oplus x_2x_3 \oplus x_2x_4 \oplus x_3x_4 \oplus x_3x_5 \oplus x_1x_6 \oplus x_2x_6 \oplus x_3x_6 \oplus x_4x_6 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_5 \oplus x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_1x_2x_6 \oplus x_1x_3x_6 \oplus x_1x_4x_6 \oplus x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_2x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_1x_2x_3x_4x_6 \oplus x_1x_2x_4x_5x_6$	32	5
f_4	$x_2 \oplus x_4 \oplus x_6 \oplus x_1x_2 \oplus x_2x_3 \oplus x_2x_4 \oplus x_3x_4 \oplus x_1x_5 \oplus x_2x_5 \oplus x_3x_5 \oplus x_2x_6 \oplus x_4x_6 \oplus x_5x_6 \oplus x_1x_3x_4 \oplus x_2x_3x_5 \oplus x_1x_2x_6 \oplus x_1x_4x_6 \oplus x_1x_5x_6 \oplus x_3x_5x_6 \oplus x_1x_2x_3x_5 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_2x_3x_5x_6$	25	5

Table 6.46: ANF and Degree of S-Box 8 BFs.

Figure 6.7 displays the Walsh-Hadamard spectra of the S-Box 8 BFs obtained from R©. It is again assumed that the reader can compute the Walsh spectra via the relation in Equation 4.16.

```
> wh(s8r1)
 [1]   0   0   0   0  -4  -4  -4  -4   0   0   0   0  -4  12  -4  12   4 -12  -4  -4   0   0   8  -8  -4  -4   4 -12
[29]  -8  -8 -16   0   0   0   0   0  -4  -4  -4  -4 -16   0   0  16  12  12  -4  -4  12  12 -12   4  -8   8 -16 -16
[57]   4   4  12  -4 -16  16   8  -8
> wh(s8r2)
 [1]   0   0   0   0   4   4  -4  -4   0   0   0   0  20 -12  -4  -4  -8   8  -8  -8   4   4  -4  12   0  16   0   0
[29]  -4  -4   4 -12  -4   4   4  -4  -8   0   8   0  -4 -12  20  -4   8   0  -8   0   4  -4  -4 -12   8  16   8 -16
[57]  -4   4   4  12  16   8  16   8
> wh(s8r3)
 [1]   0   0   0   0  -4   4  -4   4  -8   0   8   0   4   4   4 -12   4   4   4   4  16  -8   0   8   4  -4   4  12
[29]   0  16   0   0   0   0   0   0  -4   4  -4   4  -8   0  -8 -16   4   4  20   4  -4  12  12  -4  -8  16  -8  16
[57] -20 -12  12   4  -8  -8  -8   8
> wh(s8r4)
 [1]   0   0   0   0   0  -8   0   8   4   4   4   4  -4   4  12   4   0   0   0   0 -16  -8   0  -8  -4  -4  -4  -4
[29]   4  -4   4  12   4  -4  -4   4  12  -4 -12   4  16  -8   8   0   0  16  -8   8  -4   4  20  12   4 -12  -4  12
[57]  16   8   8 -16   0 -16   8  -8
```

Figure 6.7: Walsh-Hadamard Spectra of S-Box 8 BFs.

Tables 6.47, 6.48, and 6.49 follow in the same manner as before.

Function	**Cayley Graph Spectra** ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$)	**Distinct** λ_i
f_1	$\begin{pmatrix} -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 2 & 7 & 3 & 5 & 16 & 17 & 5 & 3 & 5 & 1 \end{pmatrix}$	10
f_2	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 2 & 4 & 2 & 7 & 10 & 14 & 14 & 5 & 4 & 1 & 1 \end{pmatrix}$	11
f_3	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 10 & 32 \\ 1 & 4 & 4 & 3 & 17 & 14 & 7 & 9 & 2 & 1 & 1 & 1 \end{pmatrix}$	12
f_4	$\begin{pmatrix} -10 & -8 & -6 & -4 & -2 & 0 & 2 & 4 & 6 & 8 & 32 \\ 1 & 3 & 5 & 6 & 13 & 13 & 11 & 6 & 2 & 3 & 1 \end{pmatrix}$	11

Table 6.47: Cayley Graph Spectra of S-Box 8 BFs.

Function	**Laplacian Spectra** ($\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$)
f_1	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 \\ 1 & 5 & 3 & 5 & 17 & 16 & 5 & 3 & 7 & 2 \end{pmatrix}$
f_2	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 1 & 4 & 5 & 14 & 14 & 10 & 7 & 2 & 4 & 2 \end{pmatrix}$
f_3	$\begin{pmatrix} 0 & 22 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 1 & 1 & 2 & 9 & 7 & 14 & 17 & 3 & 4 & 4 & 1 \end{pmatrix}$
f_4	$\begin{pmatrix} 0 & 24 & 26 & 28 & 30 & 32 & 34 & 36 & 38 & 40 & 42 \\ 1 & 3 & 2 & 6 & 11 & 13 & 13 & 6 & 5 & 3 & 1 \end{pmatrix}$

Table 6.48: Laplacian Spectra of Cayley Graphs Associated with S-Box 8 BFs.

Crypto Property	f_1	f_2	f_3	f_4
Degree	5	5	5	5
Balanced	Yes	Yes	Yes	Yes
Weight	32	32	32	32
Nonlinearity	24	22	22	22
Algebraic Immunity	3	3	3	3
Correlation Immunity Order	0	0	0	0
Resiliency Order	0	0	0	0

Table 6.49: Cryptographic Properties of S-Box 8 BFs.

Spectral Observations

Table 6.50 depicts the relevant properties of the Cayley graphs associated with the S-Box 8 BFs.

Graph Parameter	Γ_{f_1}	Γ_{f_2}	Γ_{f_3}	Γ_{f_4}
Regularity; deg	Yes; 32	Yes; 32	Yes; 32	Yes; 32
Connected; $k(\Gamma_f)$	Yes; 1	Yes; 1	Yes; 1	Yes; 1
Bipartite	No	No	No	No
Rank(A_{f_i}**)**	48	50	50	51
Diameter	2	2	2	2
Spanning Trees; $\tau(\Gamma_f)$	2.2801×10^{92}	1.0980×10^{90}	1.7276×10^{93}	1.7299×10^{93}
Clique Number	8	8	8	8
Independence Number	8	8	8	8
Chromatic Number	8	8	8	8

Table 6.50: Properties of Cayley Graphs Associated with S-Box 8 BFs.

6.3 Relations

The following observed relations are specific to the DES S-Box BFs and their associated Cayley graphs. These should not be universalized to all BFs used in similar substitution steps within a cryptosystem.

1. The constant term 1 appears in the ANF of a BF if and only if the associated Cayley

graph has a loop at every vertex.

2. The functions within an S-Box with the smallest number of terms in their ANF also have the smallest number of degree 5 terms.
3. Within the same S-Box, if multiple Cayley graphs have the same set of eigenvalues, then their corresponding BFs have the same nonlinearity. Furthermore, this nonlinearity is 22.
4. The function(s) with the highest nonlinearity also have the smallest number of distinct eigenvalues when compared to other functions within the same S-Box; similarly, the function(s) with the lowest nonlinearity also have the largest number of distinct eigenvalues.
5. Of the 32 total functions, seven achieve the maximum nonlinearity of 24. These seven functions as graphs do not contain ± 10 as eigenvalues.
6. Six of the 32 total functions achieve a nonlinearity of 22. These functions as graphs do not have ± 12 as eigenvalues. Furthermore, these functions have at most 31 terms in their ANF. The functions with nonlinearity 22 also have the largest number of distinct eigenvalues when compared to other functions within the same S-Box.
7. A function achieves the minimum nonlinearity of 20 if and only if $\lambda_i \in \{\pm 12\}$.
8. The Cayley graph with the largest multiplicity of 0 as an eigenvalue in each S-Box also has an adjacency matrix $\boldsymbol{A}$ with the smallest rank. Furthermore, if two or more Cayley graphs within the same S-Box have the same multiplicity of 0 as an eigenvalue, then their corresponding adjacency matrices have the same rank.
9. There is no observed pattern in the number of spanning trees in the Cayley graphs. This is somewhat interesting since all of the graphs are 32-regular, and have the same diameter, chromatic number, independence number, and clique number.
10. S-Box 2 is the only box to use a BF with algebraic degree four. Surprisingly, this function achieves the maximum nonlinearity of 24 and its Cayley graph has the smallest number of distinct eigenvalues across all S-Boxes.
11. Beginning with S-Box 3, at least two functions within each box have the same set of eigenvalues.
12. S-Box 4 is rather interesting with regards to the Cayley spectrum. Hellman and Davio noted the redundancy in this S-Box, sparking many to believe that this box was the trap door left behind by the designers. All four BFs in the fourth S-Box have the

same nonlinearity, the same set of Cayley eigenvalues, and their adjacency matrices all have the same rank. Granted, the ANFs are different, but the second and third functions have Cayley graphs with the exact same spectra.

13. The set of possible nonlinearity values $\{20, 22, 24\}$ is the same as the set of *spectral gap*[12] values. Furthermore, for S-Boxes 4-7, these two values are equal.

6.4 Expanders

Recall in Subsection 5.3.2 we introduced the Cheeger constant with respect to cuts in a graph. Another application of connectivity deals with the *expander graph*. The expander graph is a regular graph (typically of small degree) such that the number of neighbors of any subset of the vertex set containing at most half of the total nodes is at least a constant factor of its size [85]. More formally, an ε-expander is a regular graph $G = (V, E)$ such that for every set $S \subset V$ with $|S| \leq \frac{|V|}{2}$, the number of nodes in $V \backslash S$ adjacent to some $x \in S$ is at least $\varepsilon|S|$. If the spectral gap for a r-regular graph is at least $2\varepsilon r$, then the graph is an ε-expander [85]. Also [104], an r-regular graph is an ε-expander if the Cheeger constant, h_G is at least ε, i.e., $h_G \geq \varepsilon$. Hence, the term *expansion* is closely related with cuts (vertex, edge, spectral, etc.). Since expander graphs exhibit strong connectivity properties, they are often sought out in many computer based algorithms.

Expanders have wide applications, especially in computer science and the design of communication networks. Expander graphs were first defined in the 1970s [105] by Leonid Bassalygo and Michael Pinsker. It is generally difficult to construct an expander graph from scratch, since they are simultaneously sparse and highly connected. Thus, much of the work dealing with these graphs is theoretical in nature. However, random graphs often make good expanders, and we have multiple construction methods to do this. Expander graphs also have application in error correcting codes as well as pseudorandom numbers.

Construction of r-regular expanders implies control of the spectral gap, denoted from now on as $\lambda = r - \lambda_{n-1}$. Cheeger and Peter Buser bounded the Cheeger constant in terms of the spectral gap as

$$\frac{\lambda}{2} \leq h_G \leq \sqrt{2r\lambda}.$$

[12] The *spectral gap* is defined to be the difference between the largest and second largest eigenvalue, i.e., $\lambda_n - \lambda_{n-1}$. See Section 6.4.

The question remains how large the spectral gap can be. This question obviously relies on the value for λ_{n-1}, and by the bounds on the Cheeger constant we see that a large spectral gap implies high expansion. Alon and Ravi Boppana showed that this gap could be expressed by bounding the second largest eigenvalue. In particular,

$$\lambda_{n-1} \geq 2\sqrt{r-1} - o_n(1),$$

where the term $o_n(1)$ tends to zero as n becomes large [105]. This term is simplified from a fractional ratio of a constant and the diameter of a graph. The interesting case occurs when this inequality is not satisfied.

Alexander Lubotzky et al. [89] coined the term *Ramanujan graph* for an r-regular graph in which the largest eigenvalue other than $\lambda_n = r$ is less than or equal to the Alon-Boppana bound. Ramanujan graphs are named after Indian mathematician Srinivasa Ramanujan, and because they achieve close to the largest spectral gap possible, Ramanujan graphs give good explicit constructions for expanders; they are often considered to be the most well-connected among regular graphs. Precisely, let G be an r-regular graph and let $\lambda(G)$ be $\max_{|\lambda_i|<r} |\lambda_i|$. Then G is *Ramanujan* if $\lambda(G) \leq 2\sqrt{r-1}$. Interestingly, Lubotzky et al. constructed their Ramanujan graphs from Cayley graphs; the Petersen graph is an example of a Ramanujan graph. As a consequence, most constructions of Ramanujan graphs are algebraic in nature. Ramanujan graphs have an interesting niche in coding theory; certain codes such as Robert Gallager's Low Density Parity Check Codes can be constructed using Ramanujan graphs [106]. Since these graphs are good examples of connectivity, a *family* of Ramanujan graphs can yield a *family* of expanders.

While the literature varies about loop inclusion, Table 6.51 includes the DES Boolean Cayley graphs that satisfy the Ramanujan property, namely $\lambda \leq 2\sqrt{32-1} \approx 11.13552873$. If loops are included, then 26 out of the 32 Cayley graphs are Ramanujan. A star (*) indicates that the corresponding Cayley graph has loops. Given the large number of Ramanujan graphs in Table 6.51 out of the 32 possible, perhaps this yields important design considerations about S-Box construction using BFs. Interestingly, the six graphs that are not Ramanujan are also the only ones in which the associated BFs achieve the smallest nonlinearity of 20.

S-Box	Ramanujan
S_1	f_4^*
S_2	f_2, f_3, f_4^*
S_3	f_1^*, f_3^*
S_4	f_1, f_2^*, f_3^*, f_4
S_5	f_1, f_2^*, f_3, f_4^*
S_6	f_1^*, f_2^*, f_3^*, f_4
S_7	f_1, f_2^*, f_3, f_4
S_8	f_1^*, f_2, f_3, f_4

Table 6.51: The DES Functions with Ramanujan Cayley Graphs.

6.5 Distance to Linear Functions

An interesting application of nonlinearity involves finding the nearest linear or affine function to a BF. Recall the WHT given by

$$\hat{F}(\boldsymbol{u}) = W(\hat{f})(\boldsymbol{u}) = \sum_{\boldsymbol{x}\in\mathbb{F}_2^n} (-1)^{f(\boldsymbol{x})\oplus<\boldsymbol{u}\cdot\boldsymbol{x}>}.$$

This equation is also equal to the number of 0s minus the number of 1s in the function $f \oplus \ell_{\boldsymbol{u}}$, where $\ell_{\boldsymbol{u}}$ is the linear function $\ell_{\boldsymbol{u}}(\boldsymbol{v})$. Thus, $W(\hat{f})(\boldsymbol{u}) = 2^n - 2wt(f \oplus \ell_{\boldsymbol{u}}) = 2^n - 2d(f, \ell_{\boldsymbol{u}})$. It follows that for a function f and a fixed linear function $\ell_{\boldsymbol{u}}(\boldsymbol{v})$, we have

$$d(f, \ell_{\boldsymbol{u}}) = \frac{1}{2}(2^n - W(\hat{f})(\boldsymbol{u})). \tag{6.1}$$

Equation 6.1 implies that the nearest affine function $\ell_{\boldsymbol{u},a_0}(\boldsymbol{v}) = a_0 \oplus \langle \boldsymbol{u}, \boldsymbol{v} \rangle, a_0 \in \mathbb{F}_2$, to f (in terms of Hamming distance) is the function where $|W(\hat{f})(\boldsymbol{u})|$ is the largest [39]. We give an example of how to find the nearest affine function to the first S-Box BF, and then the remaining functions are merely listed.

First recall that the nonlinearity of f_1 in S-Box 1 is 20, i.e., $\mathscr{N}_{f_1} = 20$. The largest Walsh-Hadamard (absolute) value of this function is 24, which occurs for the input vec-

tor $\alpha_{43} = 101011$. To find the nearest affine function, we compute

$$\begin{aligned}
\ell_{\boldsymbol{u},a_0}(\boldsymbol{v}) &= a_0 \oplus \langle \boldsymbol{u}, \boldsymbol{v} \rangle \\
\ell_{\alpha_{43},1}(\boldsymbol{v}) &= 1 \oplus \langle 101011, \boldsymbol{v} \rangle \\
&= 1 \oplus (101011) \cdot (x_6, x_5, x_4, x_3, x_2, x_1) \\
&= 1 \oplus x_1 \oplus x_2 \oplus x_4 \oplus x_6.
\end{aligned}$$

As a check, we can see that $d(f_1, \ell_{\alpha_{43},1}) = \frac{1}{2}(2^6 - 24) = 20$, which matches the nonlinearity of f_1. Thus, we need to change 20 bits in f_1 in order to arrive at the affine function $1 \oplus x_1 \oplus x_2 \oplus x_4 \oplus x_6$. It should also be noted that some of the DES functions have multiple vectors which yield the largest WHT value, e.g., f_4 in S-Box 1 has eight vectors that produce ± 16. For these such functions, we only list one possible affine function. Table 6.52 lists the nearest affine functions to the DES S-Box functions.

S-Box	Function	$\mathscr{N}_{f_i}$	α	Nearest Affine Function
1	f_2	20	54	$x_2 \oplus x_3 \oplus x_5 \oplus x_6$
1	f_3	20	41	$x_1 \oplus x_4 \oplus x_6$
1	f_4	24	53	$1 \oplus x_1 \oplus x_3 \oplus x_5 \oplus x_6$
2	f_1	20	47	$1 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_6$
2	f_2	24	15	$x_1 \oplus x_2 \oplus x_3 \oplus x_4$
2	f_3	22	21	$x_1 \oplus x_3 \oplus x_5$
2	f_4	24	61	$1 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6$
3	f_1	22	49	$1 \oplus x_1 \oplus x_5 \oplus x_6$
3	f_2	20	54	$1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6$
3	f_3	22	29	$1 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_5$
3	f_4	20	28	$x_3 \oplus x_4 \oplus x_5$
4	f_1	22	14	$x_2 \oplus x_3 \oplus x_4$
4	f_2	22	30	$1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5$
4	f_3	22	14	$1 \oplus x_2 \oplus x_3 \oplus x_4$
4	f_4	22	13	$x_1 \oplus x_3 \oplus x_4$
5	f_1	22	20	$x_3 \oplus x_5$
5	f_2	24	46	$1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_6$
5	f_3	24	42	$x_2 \oplus x_4 \oplus x_6$
5	f_4	22	52	$1 \oplus x_3 \oplus x_5 \oplus x_6$
6	f_1	22	31	$1 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5$
6	f_2	22	29	$1 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_5$
6	f_3	22	55	$1 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6$
6	f_4	22	41	$x_1 \oplus x_4 \oplus x_6$
7	f_1	22	46	$x_2 \oplus x_3 \oplus x_4 \oplus x_6$
7	f_2	22	20	$1 \oplus x_3 \oplus x_5$
7	f_3	24	40	$x_4 \oplus x_6$
7	f_4	22	62	$x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6$
8	f_1	24	43	$1 \oplus x_1 \oplus x_2 \oplus x_4 \oplus x_6$
8	f_2	22	12	$x_3 \oplus x_4$
8	f_3	22	56	$x_4 \oplus x_5 \oplus x_6$
8	f_4	22	50	$x_2 \oplus x_5 \oplus x_6$

Table 6.52: The Nearest Affine Functions to the DES S-Box BFs.

CHAPTER 7:
Extensions on DES Substitution Boxes

Recall that Adams and Tavares [50] explained that good BFs used in S-Boxes need to satisfy the SAC. Granted, the SAC did not exist at the time that DES was introduced, and Webster and Tavares [58] even demonstrated that the DES S-Boxes do not satisfy the SAC. In this chapter, we analyze one of the design criteria of the DES S-Boxes and apply it to the coordinate vectorial BFs.

7.1 Methods

The specific design criteria we examine is listed by Coppersmith [21] as property (S-5), i.e., by complementing the middle two input bits, we should see the output bits differing in at least two positions. Mathematically, the DES S-Boxes are required to adhere to the following: $f(\boldsymbol{x})$ and $f(\boldsymbol{x} \oplus 001100)$ differ in at least two bits. This criterion was based on the S-Box as a function, i.e., $f : \mathbb{F}_2^6 \to \mathbb{F}_2^4$. We cannot specifically examine this property on the coordinate BFs because our outputs are single bits rather than strings of four bits. Thus, we perform a PC(2) check on the coordinate functions using Coppersmith's vector 001100. We aim to answer the following questions in this chapter:

1. Do the DES S-Box coordinate functions satisfy the PC of degree 2?
2. Do the DES S-Box coordinate functions satisfy the PC of degree 1, i.e., SAC?

Recall that for a function to satisfy the PC of degree $k = 2$, we need to check all possible two-bit changes in the inputs and verify that the output changes in exactly one half of the total outputs. Also recall that this can be done by either counting the number of positions where $f(\boldsymbol{x})$ and $f(\boldsymbol{x} \oplus \boldsymbol{a})$ differ, or by verifying that the weight of $f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus \boldsymbol{a}) = 2^{n-1}$. If $wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100)) \neq 32$ for any function f_i in the DES S-Boxes, $1 \leq i \leq 32$, then we can conclude that f_i does not satisfy PC(2).

We already know that the DES S-Boxes do not satisfy the SAC, but this does not imply that the row functions do not satisfy this property. We aim to shed light on this concept in this chapter.

7.2 Results on Propagation Criteria of Degree 2

Tables 7.1, 7.2, 7.3, and 7.4 display the results of the PC(2) check for the vector 001100. If a row is highlighted in green, then it satisfies the check for this vector; all others are eliminated from the check.

S-Box 1	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	36
f_2	32
f_3	36
f_4	32

S-Box 2	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	24
f_2	24
f_3	32
f_4	32

Table 7.1: Results of PC(2) Check on S-Boxes 1 and 2.

S-Box 3	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	28
f_2	32
f_3	24
f_4	32

S-Box 4	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	28
f_2	28
f_3	28
f_4	28

Table 7.2: Results of PC(2) Check on S-Boxes 3 and 4.

S-Box 5	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	28
f_2	36
f_3	32
f_4	36

S-Box 6	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	28
f_2	24
f_3	28
f_4	32

Table 7.3: Results of PC(2) Check on S-Boxes 5 and 6.

S-Box 7	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	24
f_2	32
f_3	28
f_4	32

S-Box 8	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 001100))$
f_1	28
f_2	32
f_3	40
f_4	36

Table 7.4: Results of PC(2) Check on S-Boxes 7 and 8.

For these 11 functions that are still eligible to satisfy PC(2), eight are further eliminated with a check on the vector $\boldsymbol{a} = 110000$. The final three are also eliminated with checks on vectors $\boldsymbol{b} = 101000$ and $\boldsymbol{c} = 100100$. Therefore, we reach the following conclusion concerning PC.

Result 1: The 32 coordinate BFs comprising the DES S-Boxes **do not** satisfy PC(2).

7.3 Results on Strict Avalanche Criteria

In this section, we display the results of the SAC check on the DES S-Box coordinate functions. Table 7.5 depicts the check of SAC using the vector $\boldsymbol{a} = 100000$.

S-Boxes 1-4	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 100000))$
S_1	48
	44
	48
	40
S_2	36
	44
	44
	40
S_3	44
	52
	40
	36
S_4	48
	36
	48
	36

S-Boxes 5-8	
f_i	$wt(f(\boldsymbol{x}) \oplus f(\boldsymbol{x} \oplus 100000))$
S_5	36
	48
	44
	48
S_6	44
	40
	40
	48
S_7	44
	36
	48
	48
S_8	44
	40
	44
	40

Table 7.5: Results of SAC Check on DES S-Boxes.

Note that there are no functions in Table 7.5 with a corresponding weight of 32 in the second column. Since none of these functions have this property, there is no need to check any other vector of weight one in $\mathbb{F}_2^6$. Therefore, we reach the following conclusion:

Result 2: The 32 coordinate BFs comprising the DES S-Boxes **do not** satisfy PC(1), i.e., SAC. Furthermore, we are justified in stating the implication from Webster and Tavares (only for DES). If the S-Box function $f : \mathbb{F}_2^6 \to \mathbb{F}_2^4$ does not satisfy the SAC, then its coordinate BFs do not satisfy the SAC either.

CHAPTER 8:
Conclusion

In this chapter, we summarize the findings of this thesis and present some aspects requiring further research.

8.1 Summary of Results

The goal of this thesis was to analyze DES in a new light. We used techniques from spectral graph theory to make statements about the Cayley graphs associated with the DES BFs. Several loose connections were also made between the cryptographic properties of these BFs and the Cayley graph spectra.

The Cayley graphs of these BFs all seem to share many of the same graph properties, particularly in diameter, clique number, independence number, and chromatic number. Since all 32 graphs are 32-regular, however, this is not so hard to believe. Many of the cryptographic properties of the BFs are also the same, such as degree, balance, weight, algebraic immunity, correlation immunity, and resiliency. The nonlinearity of the BFs is the primary property of variance, and it seems to be related to the multiplicity of the graph eigenvalues (in the case of DES at least).

We also found a new characterization of the DES Cayley graphs as Ramanujan graphs. These are graphs with special properties in regards to expansion; expansion relies on the size of the spectral gap. Also, we confirmed that the DES BFs do not satisfy the SAC nor the PC(2).

8.2 Areas for Future Work

There are other areas that could be extended from the work of this thesis. These areas are summarized in the following list.

1. DES Related
    - What can we learn from other matrices associated with the DES BFs, e.g., normalized Laplacian, signless Laplacian, incidence, etc.?

- What can be investigated with the energy spectrum of the BFs, i.e., the square of the WT? Is there a relation between the energy spectrum and the cryptographic properties?
- Can the inverse eigenvalue problem be applied here, i.e., can we deduce information about the graph spectra from a family of matrices producing this graph?
- Can we find patterns in the number of random walks in the Cayley graphs?
- What is the energy of the Cayley graphs, i.e., the sum of the adjacency matrix eigenvalues in absolute value, and can we determine a relation with the properties of the BFs? Can we determine a formula for the energy of the Cayley graph for a BF on n variables?

2. Non-DES Related
   - Apply spectral graph theoretic techniques to other block ciphers such as AES, or even the combiner functions used in stream ciphers.
   - Investigate relations between Ramanujan graphs and BFs used in cryptosystems.
   - What more can be done with the Laplacian spectra? If we bound the Laplacian eigenvalues by known relations, how are the associated BFs affected?
   - Can we determine a general formulaic relationship between the cryptographic properties of any BF and the spectrum of its associated Cayley graph?
   - Is there a relationship between the spectral gap of a Cayley graph and the nonlinearity of its associated BF?

APPENDIX: Thesis Code

This appendix displays some of the code used from Maple to help compute some of the properties examined in this thesis. Potential users of this code should validate its execution before implementation.

A.1 Adjacency Matrix Coding

```
> restart;
Build list of 2^6 input vectors as list of sequences
> a := [seq(ListTools[Reverse]( convert(i+64,base,2)[1..-2]), i=0..63)];
Confirm list has 2^6 elements
> nops(a);
Test extraction from list
> a[12];
> a[32];
Test mod 2 addition on elements of a
> % + %% mod 2;
Assign truth table outputs to new sequence list; change as needed
> b := [1,1,1,1,1,1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,1,0,1,1,1,
0,1,0,1,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,0,0,0,0,1,1,0,1,1,0,1];
Confirm list has 2^6 elements
> nops(b);
Test to extract i-th item from list
> a[64];a[4];b[2];b[4];
Create function/mapping from set a to set b
> for i from 1 to 64 do f(a[i]) := b[i]; od;
Test the function
> f(a[2]);f(a[4]);
Test bit operations
> a[12] + a[32] mod 2;
All possible XOR elements in set a
> for i from 1 to 63 do a[1] + a[1+i] mod 2;
```

```
f(%); od; printf("break here");
for i from 1 to 62 do a[2] + a[2+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 61 do a[3] + a[3+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 60 do a[4] + a[4+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 59 do a[5] + a[5+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 58 do a[6] + a[6+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 57 do a[7] + a[7+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 56 do a[8] + a[8+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 55 do a[9] + a[9+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 54 do a[10] + a[10+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 53 do a[11] + a[11+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 52 do a[12] + a[12+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 51 do a[13] + a[13+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 50 do a[14] + a[14+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 49 do a[15] + a[15+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 48 do a[16] + a[16+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 47 do a[17] + a[17+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 46 do a[18] + a[18+i] mod 2;
```

```
f(%); od; printf("break here");
for i from 1 to 45 do a[19] + a[19+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 44 do a[20] + a[20+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 43 do a[21] + a[21+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 42 do a[22] + a[22+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 41 do a[23] + a[23+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 40 do a[24] + a[24+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 39 do a[25] + a[25+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 38 do a[26] + a[26+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 37 do a[27] + a[27+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 36 do a[28] + a[28+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 35 do a[29] + a[29+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 34 do a[30] + a[30+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 33 do a[31] + a[31+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 32 do a[32] + a[32+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 31 do a[33] + a[33+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 30 do a[34] + a[34+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 29 do a[35] + a[35+i] mod 2;
```

```
f(%); od; printf("break here");
for i from 1 to 28 do a[36] + a[36+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 27 do a[37] + a[37+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 26 do a[38] + a[38+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 25 do a[39] + a[39+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 24 do a[40] + a[40+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 23 do a[41] + a[41+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 22 do a[42] + a[42+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 21 do a[43] + a[43+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 20 do a[44] + a[44+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 19 do a[45] + a[45+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 18 do a[46] + a[46+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 17 do a[47] + a[47+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 16 do a[48] + a[48+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 15 do a[49] + a[49+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 14 do a[50] + a[50+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 13 do a[51] + a[51+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 12 do a[52] + a[52+i] mod 2;
```

```
f(%); od; printf("break here");
for i from 1 to 11 do a[53] + a[53+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 10 do a[54] + a[54+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 9 do a[55] + a[55+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 8 do a[56] + a[56+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 7 do a[57] + a[57+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 6 do a[58] + a[58+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 5 do a[59] + a[59+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 4 do a[60] + a[60+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 3 do a[61] + a[61+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 2 do a[62] + a[62+i] mod 2;
f(%); od; printf("break here");
for i from 1 to 1 do a[63] + a[63+i] mod 2;
f(%); od; printf("break here");
```

A.2 PC Check Coding

```
> restart;
> a := [seq(ListTools[Reverse]( convert(i+64,base,2)[1..-2]), i=0..63)];
Change as needed
> b := [0,0,1,0,0,0,0,1,1,1,1,0,0,1,1,1,0,1,0,0,1,0,1,0,1,0,0,0,1,1,0,
1,1,1,1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,1,1,0,1,0,1,0,1,1,0,1,0,1,1];
Confirm 2^6 entries in each list
> nops(a);nops(b);
Add vector 001100 to every element in a mod 2;
evaluate resulting sum in function list
> for i from 1 to 64 do a[i] + [0,0,1,1,0,0] mod 2; f(%); od;
```

```
Compare original function value to PC check vector value
> for i from 1 to 64 do myvec[i] := f(a[i]) + f(a[i]+[0,0,1,1,0,0] mod 2) mod 2; od;
Count # of times "1" appears-->weight of resulting vector
> numboccur(L,1);
```

List of References

[1] S.F. Florkowski. Spectral graph theory of the hypercube. M.S. thesis, Dept. App. Math., Naval Postgraduate School, Monterey, CA, December 2008.

[2] J.B. Fraleigh. *A First Course in Abstract Algebra*. Addison-Wesley, Reading, MA, seventh edition, 2003.

[3] T.W. Hungerford. *Abstract Algebra: An Introduction*. Thomson Learning, Chicago, IL, second edition, 1997.

[4] K.H. Rosen. *Discrete Mathematics and its Applications*. McGraw Hill, New York, NY, seventh edition, 2012.

[5] R. Lidl and H. Niederreiter. *Finite Fields (Encyclopedia of Mathematics and its Applications)*, volume 20. Cambridge Univ. Press, New York, 2003.

[6] U.C. Merzbach and C.B. Boyer. *A History of Mathematics*. John Wiley and Sons, Hoboken, NJ, third edition, 2011.

[7] I. Kleiner. From numbers to rings: The early history of ring theory. *Elemente der Mathematik*, 53:18–35, 1998.

[8] S.J. Leon. *Linear Algebra with Applications*. Prentice Hall, Upper Saddle River, NJ, eighth edition, 2010.

[9] B. Schneier. *Applied Cryptography*. John Wiley & Sons, New York, second edition, 1996.

[10] D.R. Stinson. *Cryptography Theory and Practice*. Chapman & Hall, New York, third edition, 2006.

[11] C.H. Meyer and S.M. Matyas. *Cryptography: A New Dimension in Computer Data Security*. John Wiley & Sons, New York, 1982.

[12] W. Trappe and L.C. Washington. *Introduction to Cryptography with Coding Theory*. Prentice Hall, Upper Saddle River, NJ, second edition, 2006.

[13] C.P. Pfleeger and S.L. Pfleeger. *Security in Computing*. Prentice Hall, Upper Saddle River, NJ, third edition, 2003.

[14] Top 500 supercomputer sites. "Tianhe-2 (MilkyWay-2)", 2013. `http://www.top500.org/system/177999`.

[15] "Block Ciphers". lecture notes for MA 4570, Department of App. Math., July 2013.

[16] L. Brown. Cryptography and Network Security, Chapter 3, 2006. `https://app.box.com/shared/h164at4gsc`.

[17] Information security stack exchange. "Why does aes encryption take more time than decryption?", 2013. `http://security.stackexchange.com/questions/38055/why-does-aes-encryption-take-more-time-than-decryption`.

[18] Ç.K. Koç. CS178 Introduction to Cryptography, 2011. `http://cs.ucsb.edu/~koc/cs178/docs/05-modes/`.

[19] R.R. Jueneman. Analysis of certain aspects of output-feedback mode. In D. Chaum, editor, *Advances in Cryptology: Proceedings of CRYPTO 82*, pp. 99–127. Perseus, 1983.

[20] J.Y. Chouinard. Notes on the data encryption standard (DES), Sept. 2002. `http://www.csi.uottawa.ca/~chouinar/Handout_CSI4138_DES_2002.pdf`.

[21] D. Coppersmith. The data encryption standard (DES) and its strength against attacks. *IBM J. Res. & Dev.*, 38:243–250, May 1994.

[22] A. Kak. Lecture 3: Block ciphers and the data encryption standard, February 2013. `https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture3.pdf`.

[23] W.G. Barker. *Introduction to the analysis of the data encryption standard (DES)*, volume 55 of *A Cryptographic Series*. Aegean Park Press, Laguna Hills, CA, 1991.

[24] Data encryption standard. Federal Information Processing Standards Publication 46, January 1977.

[25] Data encryption standard. Federal Information Processing Standards Publication 46-2, January 1988.

[26] Data encryption standard. Federal Information Processing Standards Publication 46-3, October 1999.

[27] E. F. Brickell et al. Structure in the S-Boxes of the DES. In *Adv. in Crypt. - CRYPTO '86*, volume 263, pp. 3–8, 1987.

[28] E. Biham and A. Shamir. *Differential cryptanalysis of the data encryption standard*. Springer-Verlag, New York, 1993.

[29] G.J. Simmons. *Contemporary cryptology*. IEEE Press, New York, 1992.

[30] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. *J. CRYPT.*, 4(1):3–72, 1991.

[31] M. Bellare and P. Rogaway. Block ciphers. Lecture Notes, March 2014. `cseweb.ucsd.edu/users/mihir/cse207/s-bc.pdf`.

[32] M. Matsui. Linear cryptanalysis method for DES cipher. *Adv. in Crypt.—EUROCRYPT'93*, 765:386–397, 1994.

[33] M. Matsui. The first experimental cryptanalysis of the data encryption standard. *Adv. in Crypt.—CRYPTO'94*, 839:1–11, 1994.

[34] P. Junod. Linear cryptanalysis of DES. M.s. thesis, dept. c.s., École Polytechnique Fédŕale de Lausanne, Lausanne, Switzerland, 2000. `http://crypto.junod.info/lincrypt.pdf`.

[35] M. Davio et al. Analytical characteristics of the DES. In *Advances in Cryptology*, pp. 171–202. Springer, 1987.

[36] Y. Desmedt et al. Dependence of output on input in DES: small avalanche characteristics. *Advances in Cryptology—CRYPTO'84*, 196:359–376, 1985.

[37] A. Shamir. On the security of DES. *Advances in Cryptology—CRYPTO'85*, 218:280–281, 1986.

[38] S. Burris. George Boole. The Stanford Encyclopedia of Philosophy, 2014. http://plato.stanford.edu/archives/sum2014/entries/boole/.

[39] T.W. Cusick and P. Stănică. *Cryptographic boolean functions and applications*. Academic Press, San Diego, CA, first edition, 2009.

[40] C. Carlet. Boolean functions for cryptography and error correcting codes. In Y. Crama and P. L. Hammer, editors, *Boolean models and methods in mathematics, computer science, and engineering*, chapter 8, pp. 257–397. Cambridge Univ. Press, New York, first edition, 2010.

[41] A. Bernasconi and B. Codenotti. Spectral analysis of boolean functions as a graph eigenvalue problem. *IEEE transactions on computers*, 48:345–351, 1999.

[42] W. Meier et al. Algebraic attacks and decomposition of boolean functions. In C. Cachin and J. Camenisch, editors, *Adv. in Crypt.-EUROCRYPT 2004*, pp. 474–491. Springer, New York, 2004.

[43] J.L. Shafer. An analysis of bent function properties using the transeunt triangle and the SRC-6 reconfigurable computer. M.S. thesis, Dept. E. Eng., Naval Postgraduate School, Monterey, CA, September 2009.

[44] J.L. Shafer et al. Enumeration of Bent Boolean Functions by Reconfigurable Computer. In *2010 18th IEEE Ann. Int. Symp. on Field-Prog. Cust. Comp. Mach. (FCCM)*, pp. 265–272, May 2010.

[45] C. Carlet. On cryptographic complexity of boolean functions. In G. L. Mullen et al., editors, *Finite Fields with Applications to Coding Theory, Cryptography, and Related Areas*, chapter 4, pp. 53–69. Springer, New York, first edition, 2002.

[46] A. Braeken. *Cryptographic properties of boolean functions and s-boxes*. PhD thesis, Univ. of Leuven—KU Leuven, Leuven, Belgium, 2006. http://www.cosic.esat.kuleuven.be/publications/thesis-129.pdf.

[47] P. Stănică and S.H. Sung. Boolean functions with five controllable cryptographic properties. *Designs, Codes and Cryptography*, 31(2):147–157, 2004.

[48] C. Carlet. Vectorial boolean functions for cryptography. In Y. Crama and P. L. Hammer, editors, *Boolean models and methods in mathematics, computer science, and engineering*, chapter 9, pp. 398–472. Cambridge Univ. Press, New York, first edition, 2010.

[49] K. Kim et al. On generating cryptographically desirable substitutions. *IEICE transactions*, 73(7):1031–1035, February 1990.

[50] C. Adams and S. Tavares. The structured design of cryptographically good s-boxes. *Journal of cryptology*, 3(1):27–41, 1990.

[51] W. Meier and O. Staffelbach. Nonlinearity criteria for cryptographic functions. In J. J. Quisquater and J. Vandewalle, editors, *Advances in cryptology —EUROCRYPT '89*, volume 434 of *Lecture notes in computer science*, pp. 549–562. Springer, Berlin, Germany, 1990.

[52] T. Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE transactions on information theory*, 30(5):776–780, September 1984.

[53] N. Petrakos. Cube-type algebraic attacks on wireless encryption protocols. M.S. thesis, Dept. C. S., Naval Postgraduate School, Monterey, CA, September 2009.

[54] B. Chor et al. The bit extraction problem or t-resilient functions. In *Proc. of the 26th Ann. Symp. on Found. of Comp. Sci.*, pp. 396–407. Washington D.C., 1985.

[55] N.T. Courtois. Higher order correlation attacks, XL algorithm and cryptanalysis of toyocrypt. In P. Lee and C. Lim, editors, *Inf. sec. and crypt. —ICISC 2002*, volume 2587 of *Lecture notes in computer science*, pp. 182–199. Springer, Berlin, Germany, 2003.

[56] N.T. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In E. Biham, editor, *Adv. in crypt. —EUROCRYPT 2003*, volume 2656 of *Lecture notes in computer science*, pp. 345–359. Springer, Berlin, Germany, 2003.

[57] H. Feistel. Cryptography and computer privacy. *Scientific american*, 2280(5):15–23, May 1973.

[58] A.F. Webster and S.E. Tavares. On the design of s-boxes. In H. C. Williams, editor, *Adv. in cryp. —CRYPTO '85*, volume 218 of *Lecture notes in computer science*, pp. 523–534. Springer, Berlin, Germany, 1986.

[59] K. Kim. Construction of DES-like s-boxes based on boolean functions satisfying the SAC. In H. Imai et al., editors, *Adv. in crypt. —ASIACRYPT '91*, volume 793 of *Lecture notes in computer science*, pp. 59–72. Springer, Berlin, Germany, 1993.

[60] K. Kim et al. A recursive construction method of s-boxes satisfying strict avalanche criterion. In A. J. Menezes and S. A. Vanstone, editors, *Adv. in crypt. —CRYPTO '90*, volume 537 of *Lecture notes in computer science*, pp. 565–574. Springer, Berlin, Germany, 1991.

[61] B. Preneel et al. Propagation characteristics of boolean functions. In I. Damgard, editor, *Adv. in crypt. —EUROCRYPT '90*, volume 473 of *Lecture notes in computer science*, pp. 161–173. Springer, Berlin, Germany, 1990.

[62] L.R. Knudsen. Truncated and higher order differentials. In B. Preneel, editor, *Fast soft. encr.*, volume 1008 of *Lecture notes in computer science*, pp. 196–211. Springer, Berlin, Germany, 1995.

[63] C. Carlet. On the algebraic thickness and non-normality of Boolean functions. In *IEEE information theory workshop, 2003*, pp. 147–150. Paris, France, April 2003.

[64] C. Carlet. On the degree, nonlinearity, algebraic thickness, and nonnormality of boolean functions, with developments on symmetric functions. *IEEE transactions on information theory*, 50(9):2178–2185, September 2004.

[65] X. Zhang and Y. Zheng. GAC—the criterion for global avalanche characteristics of cryptographic functions. *J. Univ. Comp. Sci.*, 1(5):320–337, May 1995.

[66] M. Zhang and A. Chan. Maximum correlation analysis of nonlinear s-boxes in stream ciphers. In M. Bellare, editor, *Adv. in crypt. —CRYPTO 2000*, volume 1880 of *Lecture notes in computer science*, pp. 501–514. Springer, Berlin, Germany, 2000.

[67] X. Zhang and Y. Zheng. The nonhomomorphicity of boolean functions. In S. Tavares and H. Meijer, editors, *Selected areas in crypt.*, volume 1556 of *Lecture notes in computer science*, pp. 280–295. Springer, Berlin, Germany, 1999.

[68] O.S. Rothaus. On bent functions. *J. of Combinatorial Theory*, 20:300–305, 1976.

[69] R.L. McFarland. A family of difference sets in non-cyclic groups. *J. of Combinatorial Theory*, 15:1–10, 1973.

[70] J. Dillon. *Elementary hadamard difference sets*. PhD thesis, Univ. of Maryland, College Park, 1974.

[71] P. Langevin et al. On the number of bent functions with 8 variables, 2006. `http://www.liafa.jussieu.fr/~yunes/bfca/bfca06/slides/langevin-rabizonni-veron-zanotti.pdf`.

[72] A. Bernasconi et al. On the Fourier Analysis of Boolean Functions, 1996. Preprint.

[73] R. Forré. The strict avalanche criterion: spectral properties of boolean functions and an extended definition. In S. Goldwasser, editor, *Advances in cryptology —CRYPTO '88*, volume 403 of *Lecture notes in computer science*, pp. 450–468. Springer, Berlin, Germany, 1990.

[74] G. Chartrand and P. Zhang. *A First Course in Graph Theory*. Dover, Mineola, NY, 2012.

[75] L.R. Foulds. *Graph Theory Applications*, volume 1 of *Universitext*. Springer, New York, 1992.

[76] N. Biggs. *Algebraic Graph Theory*. Cambridge Univ. Press, New York, 1974.

[77] D.M. Cvetkovíc et al. *Spectra of Graphs*. Academic Press, New York, 1979.

[78] S.K. Butler. personal correspondence, 2014.

[79] F.R.K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. American Mathematical Society, Providence, RI, second edition, 1997.

[80] B. Mohar. The Laplacian Spectrum of Graphs. In Y. Alavi et al., editors, *Graph Th., Combin., and App.*, volume 2, pp. 871–898, 1991.

[81] A.E. Brouwer and W.H. Haemers. *Spectra of Graphs*. Universitext. Springer, New York, 2012.

[82] C. Godsil and G. Royle. *Algebraic Graph Theory*, volume 207 of *Graduate Texts in Mathematics*. Springer, New York, 2001.

[83] D. Cvetkovíc et al. *An Introduction to the Theory of Graph Spectra*, volume 75 of *London Mathematical Society Student Texts*. Cambridge Univ. Press, New York, 2010.

[84] K.C. Das. The Laplacian Spectrum of a Graph. *Comp. and Math. with App.*, 48:715–724, 2004.

[85] L. Lovász. Eigenvalues of Graphs, November 2007. http://www.cs.elte.hu/~lovasz/eigenvals-x.pdf.

[86] P.V. Mieghem. *Graph Spectra for Complex Networks*. Cambridge Univ. Press, New York, 2011.

[87] E.K. Çetinkaya et al. Topology Connectivity Analysis of Internet Infrastructure Using Graph Spectra. In *4th Int. Congr. on Ultra Mod. Tele. and Cont. Sys. and Work. (ICUMT)*, pp. 752–758, October 2012.

[88] A. Banerjee and J. Jost. Spectral Characterization of Network Structures and Dynamics. In N. Ganguly et al., editors, *Dyn. On and Of Compl. Net.*, Modeling and Simulation in Science, Engineering and Technology, pp. 117–132. Birkhäuser Boston, 2009.

[89] A. Lubotzky et al. Ramanujan Graphs. *Combinatorica*, 8(3):261–277, 1988.

[90] M. Fielder. Algebraic Connectivity of Graphs. *Czechoslovak Mathematical Journal*, 23 (2):298–305, 1973.

[91] R. Merris. Laplacian Matrices of Graphs: A Survey. *Linear Algebra and its Applications*, 197-198:143–176, 1994.

[92] N. Alon and V. D. Milman. λ_1, Isoperimetric Inequalities for Graphs, and Superconcentrators. *J. of Combinatorial Theory*, B 38:73–88, 1985.

[93] S. Butler. Spectral Graph Theory: Cheeger Constants and Discrepancy. In *Center for Combinatorics*. Nankai, China, 2006.

[94] A. Nilli. On the Second Eigenvalue of a Graph. *Discrete Mathematics*, 91:207–210, 1991.

[95] V. Nikiforov. Some Inequalities for the Largest Eigenvalue of a Graph. *Comb. Probab. Comput.*, 11(2):179–189, March 2002.

[96] H.S. Wilf. The Eigenvalues of a Graph and its Chromatic Number. *J. London Math. Soc.*, 42:330–332, 1967.

[97] V. Nikiforov. Chromatic Number and Spectra Radius. *Linear Algebra and its Applications*, 426:810–814, 2007.

[98] S. Butler. Spectral Graph Theory: Three Common Spectra. In *Center for Combinatorics*. Nankai, China, 2006.

[99] S. Shrikhande and Bhagwandas. Duals of Incomplete Block Designs. *J. Indian Stat. Assoc.*, 3:30–37, 1965.

[100] B. Arazi. Some Properties of Hadamard Matrices Generated Recursively by Kronecker Products. *Linear Algebra and its Applications*, 25:27–39, 1979.

[101] L. Babai. Spectra of Cayley Graphs. *J. of Combinatorial Theory*, 27:180–189, 1979.

[102] L. Lovász. Spectra of Graphs with Transitive Groups. *Period. Math. Hungar.*, 6:191–196, 1975.

[103] A. Dharwadker. The Independent Set Algorithm. [Executable], 2006.

[104] J. Fox. Eigenvalues and Expanders, 2009. http://math.mit.edu/~fox/MAT307-lecture22.pdf.

[105] S. Hoory et al. Expander Graphs and their Applications. *Bulletin of the Amer. Math. Society*, 43(4):439–561, 2006.

[106] J. Rosenthal and P. O. Vontobel. Constructions of LDPC codes Using Ramanujan Graphs and Ideas from Margulis. In *in Proc. of the 38th Allerton Conf. on Comm., Contr., and Comp.*, pp. 248–257, 2000.

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California